Challenges of working with artifacts in requirements engineering and software engineering

Ghazi, Parisa ; Glinz, Martin

Abstract: When developing or evolving software systems of non-trivial size, having the requirements properly documented is a crucial success factor. The time and effort required for creating and maintaining non-code artifacts are significantly influenced by the tools with which practitioners view, navigate and edit these artifacts. This is not only true for requirements, but for any artifacts used when developing or evolving systems. However, there is not much evidence about how practitioners actually work with artifacts and how well software tools support them. Therefore, we conducted an exploratory study based on 29 interviews with software practitioners to understand the current practice of presenting and manipulating artifacts in tools, how practitioners deal with the challenges encountered, and how these challenges affect the usability of the tools used. We found that practitioners typically work with several interrelated artifacts concurrently, less than half of these artifacts can be displayed entirely on a large screen, the artifact interrelationship information is often missing, and practitioners work collaboratively on artifacts without sufficient support. We identify the existing challenges of working with artifacts and discuss existing solutions proposed addressing them. Our results contribute to the body of knowledge about how practitioners work with artifacts when developing or evolving software, the challenges they are faced with, and the attempts to address these challenges.

DOI: https://doi.org/10.1007/s00766-017-0272-z

Challenges of working with artifacts in requirements engineering and software engineering

Parisa Ghazi[1] · **Martin Glinz**[1]

Abstract When developing or evolving software systems of non-trivial size, having the requirements properly documented is a crucial success factor. The time and effort required for creating and maintaining non-code artifacts are significantly influenced by the tools with which practitioners view, navigate and edit these artifacts. This is not only true for requirements, but for any artifacts used when developing or evolving systems. However, there is not much evidence about how practitioners actually work with artifacts and how well software tools support them. Therefore, we conducted an exploratory study based on 29 interviews with software practitioners to understand the current practice of presenting and manipulating artifacts in tools, how practitioners deal with the challenges encountered, and how these challenges affect the usability of the tools used. We found that practitioners typically work with several interrelated artifacts concurrently, less than half of these artifacts can be displayed entirely on a large screen, the artifact interrelationship information is often missing, and practitioners work collaboratively on artifacts without sufficient support. We identify the existing challenges of working with artifacts and discuss existing solutions proposed addressing them. Our results contribute to the body of knowledge about how practitioners work with artifacts when developing or evolving software, the challenges they are faced with, and the attempts to address these challenges.

✉ Parisa Ghazi
ghazi@ifi.uzh.ch

Martin Glinz
glinz@ifi.uzh.ch

[1] Department of Informatics, University of Zurich, Binzmühlestrasse 14, 8050 Zurich, Switzerland

Keywords Documentation · Artifacts · Requirements engineering · Software engineering tools · Collaboration · Interview · Exploratory study

1 Introduction

Adequate documentation is essential for successful software development [54]. This is particularly true for requirements, where missing or deficient documentation may lead to developing the wrong product. As documentation consumes time and effort, it is often neglected, leading to documents which are poor or outdated [3, 56]. However, minimizing the effort for documentation does not necessarily result in time savings and less effort, since proper documentation reduces the duration of tasks, while inadequate documentation increases the risk of making mistakes and causes late discovery of the mistakes which in turn lead to a large amount of rework [64].

In requirements engineering (RE), documenting requirements is of crucial importance, regardless of the RE process or methods being used [46]. Requirements can be documented in a variety of artifacts, from comprehensive requirements specifications to user stories, use cases, glossaries or diagrams. If multiple items (textual requirements, user stories, diagrams, etc.) are stored together in one document, we consider this as one single artifact.

Requirements engineers typically use tools for manipulating (creating, editing or viewing) artifacts, which range from general-purpose tools, such as text or diagram editors, to tools specifically built for RE purposes [12]. The usability of such tools, in particular, the effectiveness, efficiency, and satisfaction with which requirements engineers can use them for viewing and editing artifacts has a

Requirements Engineering (Springer)
DOI 10.1007/s00766-017-0272-z Published online 23 May 2017
Preprint in final layout
Final publication is available at https://link.springer.com/article/10.1007/s00766-017-0272-z

strong influence on their productivity and their willingness to produce adequate requirements artifacts.

In their daily work with requirements artifacts, requirements engineers are typically confronted with four problems. (i) They have to deal with multiple artifacts concurrently, (ii) they have to work with artifacts that are too large for being displayed entirely on the screen(s) of their computers, (iii) the artifacts they work with have multiple interconnections, and (iv) they have to work collaboratively with other requirements engineers as well as with stakeholders. Problem (ii) is particularly severe for large artifacts that do not have a linear structure. Problem (iii) is aggravated by the fact that proper information about artifact interrelationships is frequently not available. Hence, adequate tool support for coping with these problems is vital.

Our research aims at providing better tool support for requirements engineers when working collaboratively with many large and interconnected artifacts. As a first step, we wanted to gain an in-depth understanding of the state of practice in this area and draw conclusions about directions for improvement. Based on this goal, we defined four objectives for the research presented in this paper:

1. Examine the properties of requirements artifacts related to information presentation and how users work with different artifacts;
2. Investigate the challenges related to information presentation that practitioners face when working with artifacts;
3. Explore what methods practitioners use to overcome the identified challenges and how effective they are; and
4. Study to which extent existing work addresses the identified challenges.

In order to learn about the actual challenges that practitioners are faced with when working with multiple artifacts and how they deal with these challenges (objectives 1–3), we designed and conducted an exploratory study where we interviewed 29 practitioners from eleven countries. After analyzing the interview data, we closed information gaps by conducting a follow-up survey.

For two reasons, we decided to include not only requirements, but all software development artifacts, except source code in our study: First, the challenges of working with artifacts are not confined to RE. Second, recruiting participants who purely work with RE artifacts or can isolate their experience with RE artifacts from other artifacts was not possible. As a result, eight out of 29 interviewees are requirements engineers (business and software analysts), while the others are software architects, software developers, software testers, and project managers. We analyzed the interview data separately for every role and found that the results were not significantly different from those obtained for all interviewees (see Sect. 3.2). Hence, we conclude that our results not only characterize how practitioners utilize tools to deal with various types of artifacts in general, but that these results are equally valid for how requirements engineers deal with requirements artifacts.

In addition to the challenges, we have also studied the workarounds that practitioners employ to deal with the challenges we have identified. We analyze how these challenges and workarounds may affect the effectiveness, efficiency, and satisfaction with which practitioners use tools to manipulate artifacts. Finally, we discuss other studies and tools that provide support for working with artifacts, especially when artifacts are large, numerous and interconnected (objective 4).

This paper is an extended version of a conference paper [21], where we presented our exploratory study with the first analysis of our findings. In this paper, we make four additional contributions:

1. We provide a deeper analysis of the interview data, resulting in several more findings;
2. We conducted a follow-up survey which provides answers to issues that were left open in our exploratory study;
3. We consolidate all our findings into a comprehensive view, analyzing how the found challenges impact the effectiveness, efficiency, and satisfaction with which the practitioners use tools to work with artifacts;
4. We discuss to which extent existing studies and tools address the identified challenges.

The remainder of this paper is organized as follows. We define some key terms in Sect. 2. In Sect. 3, we describe our research methodology. Our key findings are presented in Sect. 4. In Sect. 5, we consolidate all our findings into a comprehensive view. Existing work that addresses the challenges we found in our study is discussed in Sect. 6. Section 7 describes further related work. In Sect. 8, we explain the threats to the validity of our study and what we did to limit them. Section 9 concludes with a summary and outlook.

2 Definition of terms

In this section, we define some key terms that we will use frequently in this paper. While these terms are being used broadly in the literature and daily life, we nevertheless provide definitions to avoid any misunderstanding caused by presumptions.

2.1 Practitioner

In this study, we consider *practitioners* to be professional persons who actively contribute to the development or evolution of a software-based system. A practitioner may

possess any role such as requirements engineer, project manager, software architect, software developer, or software tester. In the study presented in this paper, we specifically aimed at practitioners who work with artifacts on a daily basis.

2.2 Artifact

In the context of this paper, an *artifact* is any kind of textual or graphical document with the exception of source code. Artifacts may be, for example, textual requirements documents, graphic models (including Unified Modeling Language (UML) diagrams), glossaries, charts, or sketches. Artifacts can have any size and granularity, ranging from comprehensive documents to user stories, use cases, bug reports or diagrams. They may use various notations and can be interconnected to other artifacts. We excluded source code for two reasons: (i) we are primarily interested in artifacts relevant to requirements engineering, and (ii) the tools used for handling source code are different from those for handling other artifacts. When multiple elements such as textual requirements, user stories or diagrams are stored together in one document, we consider that document as a single artifact. Our rationale for doing so is the fact that the whole document needs to be opened, searched and navigated even when just a single element, such as an individual user story, needs to be retrieved or edited.

2.3 Tool

Any software product which is used for creating, editing, viewing, or managing artifacts is considered to be a *tool*. We are interested in tools that allow users to work directly with artifacts, i.e., viewing, navigating and editing them.

2.4 Screen

Regardless of the tools used for dealing with artifacts, information is presented to the tool users through display devices, for example, computer monitors, built-in screens of laptops and tablets, or electronic whiteboards. The size of these devices, i.e., the available space for displaying information in a readable size, imposes limitations when working with artifacts. In this paper, we use the term *screen* to denote any device that a tool uses to present an artifact. Practitioners use screens of various sizes and may also use multiple screens of different sizes simultaneously. When analyzing our data, we use the aggregated screen size or the maximum screen size depending on our analysis purpose. The reason is explained whenever such a decision is taken. In this paper, an artifact is considered to *fit on a screen* when it is still readable after zooming out to become entirely visible. Fitting on a screen depends on the screen size, screen resolution and the eyesight of the user. In high-resolution screens, an artifact can be highly zoomed out while preserving adequate detail. However, the detail may be unreadable due to human eyesight limitations. Considering that high-resolution screens are commonly used and, with the help of accessories if needed, there is not much difference in human eyesight, the most influential factor is the screen size.

2.5 Interaction when working with artifacts

Our work aims at better understanding how requirements or software engineers work with artifacts that are stored on their computers, i.e., how they access, search, navigate or edit artifacts with the help of *tools*, using common input/output devices such as screens, keyboards, and mouses. We study the *interaction* methods and mechanism that tools provide for enabling users to perform the access, navigation, manipulation, and management actions on artifacts. Scrolling, zooming, resizing, and switching windows are some of the most frequently used interactions. We do not investigate the actual information being stored, accessed or manipulated in these artifacts.

3 Research methodology

To investigate how practitioners work with artifacts and how they deal with the interaction challenges encountered in various situations, we conducted an exploratory study [53] based on semi-structured interviews [28]. We were interested in situations such as when the artifacts do not fit on their screen, should be handled simultaneously or their relationship information is not kept well. Later we complemented the data from the interviews with a follow-up survey. In the interviews, we were able to collect quantitatively analyzable data and information about how practitioners actually deal with the challenges they encounter at the same time. The latter was made possible by asking open questions. Consequently, our dataset is partially composed of qualitative data [52]. The interview format also gave us the chance to explain the questions to the participants well enough to avoid misunderstandings and collect more accurate answers. The survey allowed us to reach a higher number of participants and fill the information needs that we encountered when analyzing the interview data.

3.1 Research questions

From the goal and the objectives that we presented in the introduction, we derived four research questions (RQ1–

RQ4). Although our research questions are framed in the context of requirements engineering artifacts, we actually studied the more general problem of the challenges encountered when working with software engineering artifacts (except source code). As explained above in the introduction, the rationale for this decision is that these interaction challenges are not confined to requirements engineering.

RQ1. *How do different interaction-related properties of artifacts, tools, and screens affect the effectiveness and efficiency of working with requirements artifacts?*

Artifacts have different properties. Different types of artifacts require different types of interaction. Working with artifacts in software tools is more challenging when the artifact has certain properties such as being larger than the screen or being connected to other artifacts so that the practitioners need to work on multiple artifacts at the same time.

Tools and screens can also have different properties that affect user interaction. The features of the tools and the size of the screens are two such properties. To have a deeper understanding of the interaction with artifacts and improve it, we decided to investigate the properties of artifacts, tools, and screens.

RQ2. *What interaction challenges do practitioners encounter when working with requirements artifacts?*

Secondly, we wanted to know the interaction challenges that practitioners are faced with when working with requirements artifacts of different properties, particularly with respect to the presentation of artifacts on the available screen(s). We encouraged the participants to tell us about all the challenges they encounter related to navigating, manipulating and managing artifacts (or any other action they perform on their artifacts).

RQ3. *What methods do practitioners use to handle the interaction challenges of working with requirements artifacts?*

Having identified the interaction challenges experienced by practitioners, we study how they try to overcome them, e.g., whether they use features of tools or improvise other workarounds.

RQ4. *Which existing solutions address the challenges of working with RE artifacts?*

Finally, after discovering the challenges of working with artifacts and the workarounds that practitioners employ to cope with them, we identify some of the solutions (e.g., techniques proposed in research or tools) that attempt to address the challenges we found in our answer to RQ2. The answer to this question provides us with an impression of how much effort has been devoted so far to solve those challenges and encourages researchers to investigate the reasons why despite of the proposed solutions the challenges still exist.

3.2 Study design

1. Initial preparation: Our semi-structured interviews were based on an interview instrument[1], which was first elaborated as a list of questions linked to the RQs and the goals of the study. The interview instrument was designed following the guidelines stated by Oates [41]. Then, it was improved in two rounds of evaluation and feedback: first, it was evaluated by a group of RE experts to discover possible ambiguities and shortcomings. Second, we conducted two pilot interviews with a researcher from the University of Zurich and a practitioner.

The interview instrument comprises four sections: (i) characterization of the companies and the interviewee, (ii) properties and types of artifacts used by the interviewees and the tools they use to handle them, (iii) challenges interviewees encounter when working with certain types of artifacts, and (iv) how they deal with these challenges.

2. Selection of participants and demographics: The 29 practitioners we interviewed can be categorized into five roles: we had eight *requirements engineers* (business and software analysts), five *software architects*, nine *software developers*, four *project managers*, and three *software testers* (Fig. 1a).

Requirements artifacts are used almost in all phases of software development. To obtain a comprehensive view of tools and challenges related to artifact creation, modification, comprehension, and management, the study cannot be constrained to a specific phase or role of software devel-

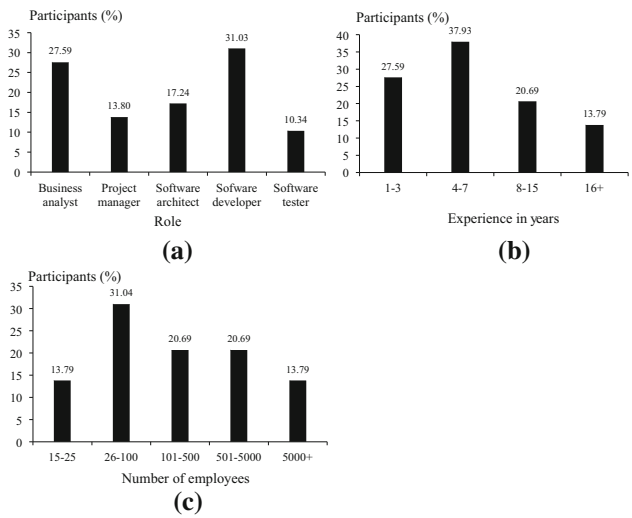

Fig. 1 Distribution of participants with regard to **a** distribution of the roles, **b** years of experience and **c** size of the company measured in number of employees

[1] http://www.ifi.uzh.ch/rerg/people/ghazi/InterviewInstrument2016.pdf

opment. Otherwise, some of the mentioned aspects will be overlooked.

Many roles in software development work with multiple types of artifacts (e.g., requirements artifacts and design artifacts). Asking participants to consider only one type of artifact when answering the questions would result in inaccurate data since it is possible that the participants unintentionally consider wrong types for some parts of the questions. Assuming that documentation is done in a similar way in different phases of software development, we neither restricted our study to a particular development role, nor asked the participants to consider specific types of documents. In the analysis phase of our study, we searched for correlations between the roles and other parts of the dataset (e.g., the size of artifacts and screens, various challenges), but did not find any. For example, Fig. 9 shows how artifact size distribution is similar among different roles. This indicates that there are no significant differences among the different phases and roles of software development with respect to the questions we are exploring.

We defined two criteria to ensure recruiting suitable participants for our study. We looked for software development practitioners who (i) had at least one year of experience of being a member of a software engineering team, and (ii) used software and requirements artifacts (textual and graphical) on a daily basis during their work. The self-evaluation of the participants' experience in working with artifacts is shown in Fig. 1b. The largest group of participants (38%) reported between four and seven years of experience and 14% had more than 16 years of experience in working with requirements and software artifacts.

With respect to company size, our study covers a wide range, from under 25 to more than 5000 employees (Fig. 1c). The largest group of participants works in companies having between 26 and 100 employees (31%).

When recruiting participants, we combined two types of sampling: snowball and convenience sampling [29]. For the snowball sampling, we started from our immediate acquaintances who were active in any phase of software development such as requirements engineering, software design, software development and testing. We sent an e-mail containing a short description of the research and the selection criteria mentioned above, making it clear that participation is voluntary. We asked them to send us a short description of their duties and whether they work with both textual and graphical artifacts or not. After each interview, we asked them to introduce other practitioners who fit the criteria. The majority of the participants believed that another interview with a person from their company would result in similar data. Although the redundant data could help us in validating the dataset gathered, we decided to invest our resources in increasing the variety of the participants, and recruit the next group of participants from the acquaintances of the first group working in other companies. The majority of our participants (86%) were recruited by snowball sampling. For the convenience sampling, we used a social network of professionals (LinkedIn). We sent LinkedIn messages (InMails) to a randomly selected set of practitioners and asked them whether they complied with our selection criteria.

Eventually, we interviewed 29 practitioners from 29 different companies, located in eleven countries from seven geographical regions, as depicted in Fig. 2.

3. Data collection and analysis: The interviews were conducted between November and December 2015. Their duration varied between 40 and 87 min, with an average of 56 min. We conducted the interviews over Skype or Google Hangouts, except for three, which were conducted face-to-face. All interviews were recorded and transcribed completely.

We used Microsoft Excel and MATLAB in the data analysis phase. We started to analyze quantitative data by first extracting the simple quantitative variables such as number of screens, and continued by computing the complex variables such as average artifact size for each participant. To analyze the qualitative data we employed a qualitative analysis approach [8] comprised of six steps: (i) preparing the raw data in spreadsheets for analysis, (ii) extracting different items from each answer of each participant, e.g., the mentioned challenges, (iii) providing a description and a code for each item, (iv) identifying the similar items by their descriptions and grouping them, (v) restructuring data based on the identified groups and counting the frequencies, and (vi) finalizing our findings.

4. Follow-up survey: Ten months after the initial interviews and the data analysis resulting in several findings, we carried out a follow-up survey to gather the information that we needed for further potential findings. Particularly, we wanted to (i) have more detailed information about storing artifact interrelationships, and (ii) know which productivity factors to consider when analyzing the challenges of handling artifacts. The survey came in two formats: as an editable pdf file[2], and as a Google Form. The participants could use one of these formats based on their

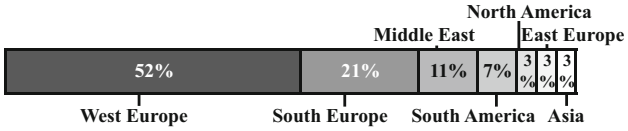

Fig. 2 Geographical distribution of participating companies

[2] http://www.ifi.uzh.ch/rerg/people/ghazi/Follow-upSurvey2016.pdf

preferences. The responses were gathered within ten days. To measure the attitude of the participants, we used a five-point Likert scale [34].

We sent the survey to 59 people and successfully received 42 responses, yielding a response rate of 71%. We made sure that all participants answered willingly. Twenty participants had also participated in the initial interviews. The remaining participants belong to two groups: (i) fourteen people whom we had identified as potential interviewees, but eventually did not interview due to their or our availability constraints, and (ii) eight persons working in academia as requirements engineering researchers whom we knew and could approach directly. Obviously, the participants in the former group meet the selection criteria we had defined for interview participants. All members of the latter group had the knowledge required for answering the survey questions. In total, as Fig. 3a shows, we received responses from academic experts (19%), software developers (28%), software architects (10%), project managers (12%), software testers (7%), and requirements engineers (24%). Among the academic experts, we had one professor, two postdoctoral researchers, three Ph.D. students and two graduate students (Fig. 3b).

All academic participants are active in the field of requirements engineering. The majority of the industry practitioners have 4–7 years of experience (41%) and 9% have more than 16 years of experience in working with artifacts (Fig. 3c). The largest group of participants are working in companies having between 15 and 100 (29.4%) employees or between 101 and 500 employees (29.4%). The remaining participants work in large companies—see Fig. 3d.

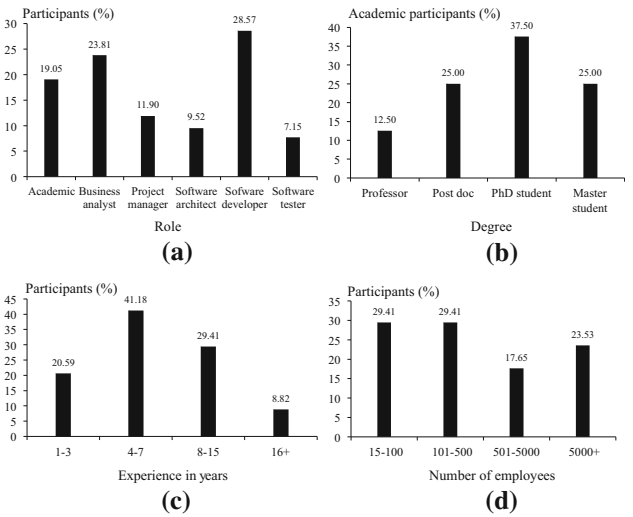

Fig. 3 Demographic information about the survey participants: **a** distribution of the roles, **b** the level of experience for academic participants measured by their academic degree, **c** years of experience in working with artifacts, **d** size of company for industry practitioners measured in number of employees

4 Key findings

In this section, we present and discuss the findings from our interviews and the follow-up survey, thus answering our research questions RQ1–RQ3. We coded our findings with respect to their corresponding research questions: FA-FC, respectively. For example, the first finding regarding RQ1 is coded FA1. For every finding, we provide the evidence from the interviews and the survey, and a short discussion of the importance of that finding. We will address RQ4 in Sect. 6.

As our interview study is based on a relatively small, non-representative sample, the quantitative figures we report in our findings must be treated with care: from a statistical viewpoint, these quantities are not generalizable. While the quantitative data we report are factual for our sample, a study based on a representative sample might yield different results. Hence, with respect to generalizability, our quantitative findings should be considered as hypotheses, rather than generally valid facts. Nevertheless, we believe that a quantitative evaluation of our study data adds value for the readers of this paper.

4.1 Interaction-related properties of artifacts (RQ1)

In a first step, we wanted to learn about the interaction-related properties of artifacts, screens, and tools. We investigated artifact properties such as the size of artifacts, the maximum number of artifacts used simultaneously and the notation used in the artifacts. We also asked about the size and the number of screens and the software tools used by the participants.

Finding FA1 Only about one-third of the graphical artifacts used by the interviewed practitioners fit on their screens.

Evidence for FA1 In the interviews, we explicitly asked the participants about the percentage of their software engineering graphical artifacts that fit on their screen according to the definition of fitting provided in Sect. 2.4. We explained to the participants to consider the situation where their artifacts are zoomed out as much as possible just before the details get too small to be recognized by their eyes. We assumed that the resolution of their screens did not limit the zooming. Figure 4 visualizes this information. In this question, we made a distinction between textual and graphical artifacts because textual artifacts have their own way of navigation, search, and management. The bounding box represents the entire set of the participant's artifacts. This space is partitioned into 29 vertical bars.

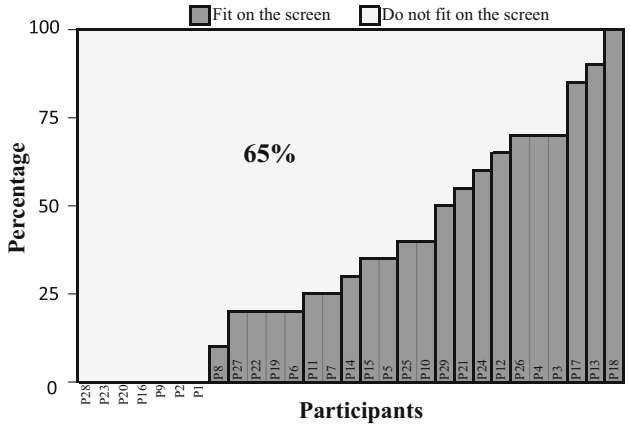

Fig. 4 Percentage of software engineering graphical artifacts that fit on the participants' screens

Each bar represents a participant and is filled according to the participant's answer. We sorted the participants based on the percentages in order to have the filled areas on one side and not-filled areas on the other side. The total not-filled area in the resulting chart (Fig. 4) represents 65% of the artifacts that do not fit on the respective participant's screens. This percentage can be ascertained by the ratio of the light gray area to the whole area of the box.

Discussion The size of artifacts affects many aspects of working with artifacts. Even when only a part of an artifact is needed for a task, a practitioner needs to search for the part of interest, navigate through the artifact to reach that part and adjust the zoom level according to the given task. Also, when practitioners modify some part of an artifact, they nevertheless need to view and consider the whole artifact in order to preserve the consistency of the artifact.

One can argue that the participants' screens have different sizes and the percentages found depend on the actual screen sizes. Although this argument is true, the chart still shows the percentage of the artifacts that are used on screens that do not permit to view these artifacts entirely. In the next finding, we have eliminated this dependency.

Finding FA2 About forty percent of the graphical artifacts do not fit on the largest screen reported in this study.

Evidence for FA2 We wanted to investigate the artifact size in a way that does not depend on the participant's screen size. The difficulty was that no common measure for artifact size exists that everybody understands and that allows comparisons. To overcome this problem, we made two decisions. (i) As the measure for artifact size, we decided to use the smallest screen size on which the artifact fit according to our definition of fitting from Sect. 2.4. (ii) Since it was nearly impossible to ask participants to describe all of their artifacts with this measure, we asked a simpler question that led us to find the distribution of their artifacts' sizes. In particular, we asked the participants to estimate, in percent, the amount of their artifacts that fit on screens of four different sizes: 11-, 15-, 22-, and 28-inch. This question is not only simpler but also provides more information.

We decided to use the estimations for 15- and 22-inch screens since, according to the frequency of screen sizes illustrated in Fig. 5, the 22-inch screen is the most used, followed by the 15-inch screen. We also used the percentages provided for 28-inch screens to include the percentage of the artifacts that fit on the largest monitor (according to this research). However, we dropped the data for 11-inch screens as only one participant used a screen of this size. The total number of screens is higher than the number of participants because many of them had more than one screen.

We visualized the data gathered from this question in the same format as Fig. 4 which resulted in finding the percentages of the artifacts that do not fit on the above-mentioned screen sizes. According to our finding, 24% of the participants' artifacts fit on a 15-inch screen, 42% of the participants' artifacts fit on a 22-inch screen, and 58% of the participants' artifacts fit on a 28-inch screen. Figure 6 shows the percentages of graphical artifacts fitting on a 28-inch screen. The bounding box represents the entire set of the participants' artifacts. This space is partitioned into 29 vertical bars. Each bar represents a participant and is filled according to the participant's percentage of artifacts that fit on a 28-inch screen. The empty space (light gray) which constitutes 41% of the whole box shows the overall percentage of the artifacts that do not fit on a 28-inch screen.

Discussion In Fig. 6, the part with the lightest gray color covers more than 41% of the area and represents the artifacts that do not fit even on the largest screen reported in this study. This result shows that supplying larger screens will alleviate the problem by fitting more artifacts but not solve the problem completely: alternative solutions are needed.

Finding FA3 More than half of the interviewed practitioners use four or more artifacts at the same time.

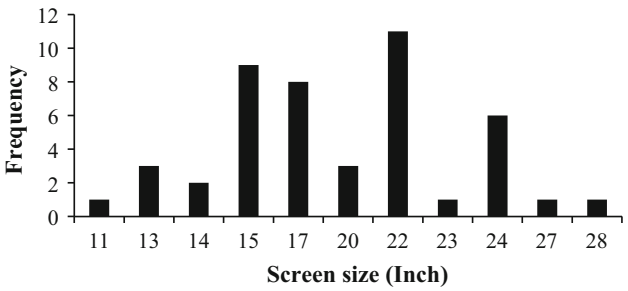

Fig. 5 Distribution of screen sizes of the participants

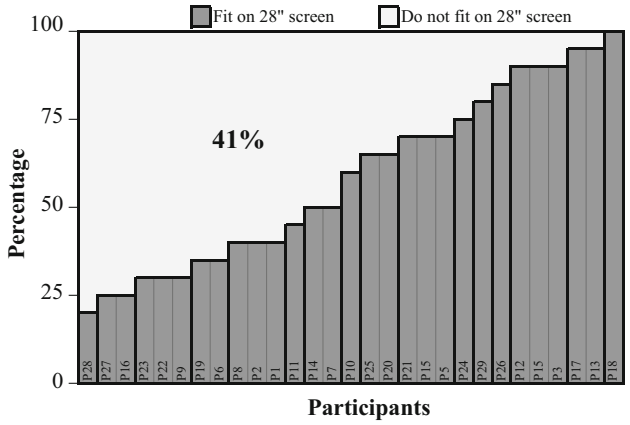

Fig. 6 Percentages of graphical artifacts that fit on a 28-inch screen

Evidence for FA3 We asked the participants about the maximum number of artifacts they use simultaneously. We received different answers ranging from one to twenty. The box plot in Fig. 7 shows the distribution of the answers. Half of the participants use between three and six artifacts at the same time; the median is four. Two practitioners with twelve and twenty artifacts used at the same time are outliers; therefore, we can say that the number of artifacts being used simultaneously ranges between one and ten. The types of the simultaneously used artifacts can be similar or different. For example, P15 was a lead business analyst working with ten artifacts at the same time of the following types: functional and nonfunctional requirements specification documents, scope specification documents and diagrams such as use case, sequence, flowchart, Business Process Model and Notation (BPMN), and mindmap.

Discussion This finding emphasizes the importance of screen space and how it is used to present artifacts. From the fact that a significant portion of artifacts are larger than the available screens (from FA1 and FA2) and more than half of the practitioners use four or more artifacts at the same time, we conclude that practitioners either need larger screens (which is limited by cost and technology) or the existing screen space must be used in a smarter way when presenting artifacts to their users. The challenges of working with multiple artifacts at the same time are discussed in FB2.

Finding FA4 More than two-thirds of the interviewed participants use customized notations for their artifacts.

Evidence for FA4 We asked our participants which notations they use for their artifacts. Except for structured text which is used by everyone, the majority of practitioners mentioned that they use the UML. Further, more than half of the participants also use customized notations. Three participants (10%) reported that they use the BPMN as well. Figure 8 displays the distribution of notations used for graphical artifacts by practitioners. The sum of the percentages is greater than 100% since many practitioners use more than one notation at the same time.

Discussion This finding shows that although UML is the dominating notation in the software development industry, companies also customize some notations based on their needs. Therefore, any assumption about the properties of artifacts (e.g., size, number, and interconnectivity) based on the UML notation or other well-known notations may result in developing non-generalizable interaction methods. Such a tool may limit its user in performing certain tasks or may force its user to perform unnecessary tasks for the custom notation chosen. This is one of the reasons why practitioners use other means (for example, pen and paper or whiteboards) to create graphic artifacts (this will be discussed in more detail in FC5).

Finding FA5 On average, practitioners use more than five applications concurrently to create and manage their artifacts.

Evidence for FA5 We asked our participants about the artifact-related applications they use and compiled a list of the applications mentioned. This list includes well-known applications as well as less popular ones. The point that caught our attention was that our participants used many applications concurrently. For instance, one of the participants mentioned ten applications that he used in each project. The tools can be used simultaneously or at different points of time. Practitioners use multiple tools at the same time when multiple artifacts are needed simultaneously and when they need multiple artifacts that require different tools to be displayed and manipulated.

Discussion According to our interviews, using multiple software tools concurrently is common in the software development industry and practitioners are used to it. In this regard, the following points may cause challenges in

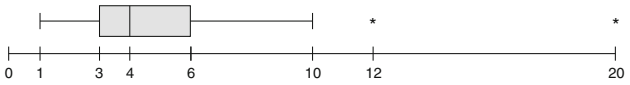

Fig. 7 Number of artifacts used at the same time

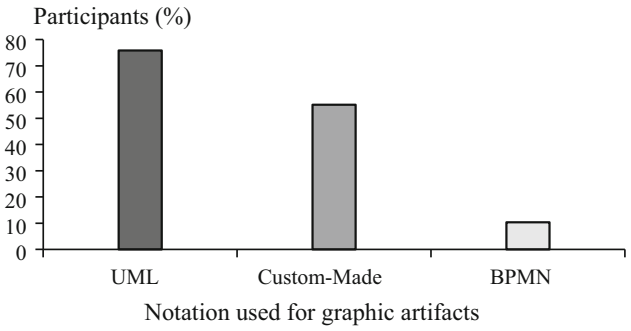

Fig. 8 The distribution of graphic notations used by participants

working with artifacts or exacerbate other challenges: (i) Practitioners must continuously switch between the applications that have different ways of interaction. The challenges of switching between windows are discussed in FB2. (ii) A new member of the team has to learn how to work with all these different applications, which causes a steep learning curve. (iii) Practitioners have to keep track of the relationships between artifacts manually, which is discussed in FC7.

Finding FA6 Although almost all participants work on their artifacts collaboratively, less than one-third of the collaboration is done with dedicated software development tools.

Evidence for FA6 Except for one of the participants, all interviewees mentioned that they collaboratively work on artifacts. According to our study, many different tools are used for allowing multiple practitioners to work on one artifact at the same time. We categorized these tools into four categories: file sharing (e.g., shared folder, Google Drive), view-only sharing (e.g., shared monitor, video conference tools), general collaboration tools (e.g., SharePoint, wiki pages) and software development-specific tools. Table 1 shows the percentages for each category and some examples. 28% of the mentioned tools are in the software development-specific category and the majority of the collaboration (45%) is done by view-only methods. 17% of the mentioned collaboration tools belong to the general collaboration tools category, e.g., Google Docs and wiki pages. The remaining 10% of the collaboration is done by simple file sharing.

Discussion The information from the interviews clearly shows that practitioners work on artifacts collaboratively. The collaboration may be a discussion about the artifact over a video conference or creating the artifact on a whiteboard. Many software development tools support collaboration directly or via plugins. However, there are many practitioners who still use methods such as sharing a file, which has major problems such as the danger of overwriting each other's work. Another group of practitioners uses general-purpose collaboration tools such as wiki pages or SharePoint, which reduce the problems of file sharing but still need extra work to import the artifacts and keep them up-to-date. This is because the editing capabilities of this type of tools are inferior to tools specifically built for the purpose of editing software development artifacts.

4.2 Challenges of working with artifacts (RQ2)

After asking the participants about the properties of their artifacts, screens, and tools, we asked about the challenges they experienced when working with artifacts larger than their screens, the challenges they faced when working with multiple artifacts at the same time, and the challenges of dealing with the interrelationships between artifacts.

Finding FB1 "Relying on memory", "Searching for information", and "Maintaining the overview" are the most important challenges in handling large artifacts.

Evidence for FB1 After gaining a perspective of the participants' artifact size and screen size, we asked them about the challenges of working with artifacts that are larger than the available screen. After gathering all challenges, we first created a comprehensive list of them. To guarantee atomic and concrete challenges, we removed general ones (e.g., "Working with large artifacts is not efficient") and dependent ones (e.g., "This type of artifact takes so much time"). Afterwards, we grouped similar challenges that were expressed in different words. For example, below we give some quotes about how participants rely on their memory. "[P23:] *It increases your cognitive overhead because you do not remember where things are*", "[P10:] *You have to imagine what is located in the part of the picture you cannot see*" and "[P14:] *Because I forget things easily I have to take notes in another place*". Table 2 presents the list of challenges and their frequency (number of participants mentioning that challenge). The calculation of the priorities is explained below.

The frequency of mentioning a challenge alone is inadequate to show the importance of the challenge, as different participants may be affected by the challenges to different extents. Therefore, we decided to rank the participants based on how heavily they are affected by the challenges of working with artifacts that are larger than their screen(s) and to use this ranking for prioritizing the challenges listed in Table 2. For this purpose, we assumed that people who have larger artifacts and smaller screens

Table 1 The percentages of different categories of collaboration tools and examples for each category

Category	Percentage (%)	Examples
View-only sharing	45	Shared monitor, video conference tools, whiteboards
Software development-specific tool	28	JIRA, Confluence, Enterprise Architect
General collaboration tool	17	SharePoint, wiki pages, Google Docs
File sharing	10	Shared folder, Google Drive

Table 2 Challenges of working with artifacts that are larger than the available screen. Prioritization is explained in the text

ID	Challenge	Priority	#Participants
C1.1	Relying on memory	1	18
C1.2	Searching for information	2	2
C1.3	Maintaining the overview	3	20
C1.4	Too much scrolling & zooming	4	29
C1.5	Knowing the current location	5	5
C1.6	Following the links	6	7

are more challenged than others since they have larger parts of their artifacts outside of their screens. For example, in such a situation, more scrolling and zooming are needed, searching is more difficult and more information is needed to be kept in the mind.

As a first step, we computed the average artifact size of each participant using the Cumulative Distribution Function (CDF). We gathered three points for the CDF of the artifact size for each participant: the participant's estimation of which percentages of artifacts fit on 15-, 22-, and 28-inch screens, respectively (according to FA2).

CDF is calculated by the following formula [38]:

$$\mathrm{CDF}(x) = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \quad (1)$$

x represents the screen size and CDF(x) is the percentage of artifacts that fit on a screen of that size. These values are known from the interviews. μ and σ are mean and variance, respectively. erf is called error function and is already known [38]. So we rewrite the formula as:

$$\sqrt{2}\,\mathrm{erf}^{-1}(2\,\mathrm{CDF}(x) - 1)\sigma + \mu = x \quad (2)$$

This is a linear equation with regard to the parameters σ and μ. Therefore, by plugging the mentioned three points, we can calculate σ and μ. In the rest of this paper, the calculated μ is called S_μ. Since it is the mean of the artifact sizes for each participant, it indicates "the size of the smallest screen that can accommodate half of the artifacts in a readable size". Figure 9 shows the overall and role-wise distribution of S_μ.

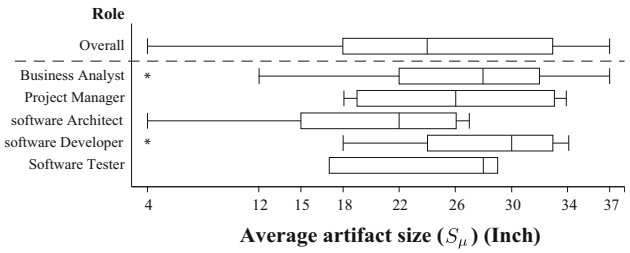

Fig. 9 Overall and rolewise distribution of the average artifact size (S_μ)

Having calculated S_μ for all participants, we can now rank them with respect to the degree to which they are affected by the challenges of working with artifacts that are larger than the available screen(s). We do this by assigning a point to each participant in a coordinate system with screen size as the x-axis and S_μ as the y-axis, and calculating the Euclidean distance of each point from a reference point which represents a hypothetical person who is more under the influence of these challenges than all others in our dataset. This hypothetical person has a screen size of 12 (less than everyone else) and a S_μ value of 40 (more than everyone else). The result is depicted in Fig. 10. The labels show the number and rank of the participants. The closer a participant is to the reference point on the top left edge, the more she or he is affected by the challenges of working with artifacts that are larger than the available screen(s). The rank of participants who have more than one screen was calculated based on their largest screen, assuming that they work with large artifacts on their largest screen.

We sorted the participants based on the calculated distances and ranked them so that the participant with the lowest distance has the highest rank of one and the participant with the highest distance has the lowest rank of 29. Then, we propagated the ranks of the participants to the challenges they mentioned. For example, if X is a challenge mentioned by three participants with ranks (1, 8, 18), we ranked X with the average of the participant ranks, which is nine. Finally, we prioritized the challenges according to their ranks as shown in Table 2.

Discussion The most frequently mentioned challenge is the problem of too much scrolling and zooming (C1.4), which is an obvious impact of working with an artifact that is larger than the screen. However, after prioritization, the

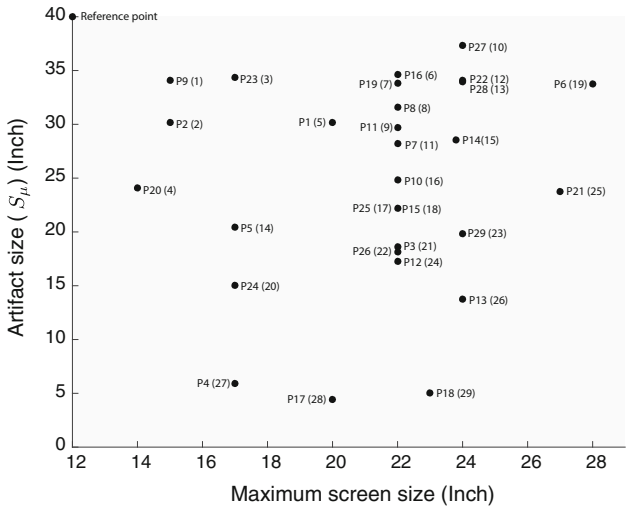

Fig. 10 Ranking of the participants with respect to artifact and screen size used. The ranks of the participants are determined by measuring their Euclidean distance from the reference point

top challenge is the participants' need to rely on their memory (C1.1), particularly to avoid excessive scrolling and zooming (C1.4). "[P27:] *I use my memory if I can avoid zooming and scrolling around*". Maintaining the overview over the artifact (C1.3) is the third top challenge with respect to the number of participants who mentioned this challenge. Interestingly, searching for information (C1.2) is the second ranked challenge, although it was mentioned by two participants only. "[P3:] *When you want to find information in an artifact and the artifact is a big one, it is very hard. Searching information in larger artifacts is harder*". This illustrates to what extent searching for information can be cumbersome when having large artifacts and a small screen.

Finding FB2 "Switching between windows" and "Working in too small windows" are the most important challenges when working with multiple artifacts.

Evidence for FB2 The challenges of working with multiple artifacts are identified and prioritized similarly to FB1 In this case, we identified three related parameters for ranking the participants: screen size, S_μ (smallest monitor size that accommodates half of the artifacts) and the number of artifacts used at the same time. We assigned a point in the three-dimensional space to each participant. We assumed that a person with large artifacts, a small screen and a high number of concurrently used artifacts is impacted more strongly by the challenges of working with multiple artifacts than others. Therefore, we calculated the Euclidean distance between each participant point and the reference point showing a hypothetical person having an extremely small screen, a S_μ value of 40 (higher than everyone else), and using 20 artifacts at the same time (the highest in our dataset). When analyzing, we aggregated the screen sizes of participants who use multiple screens, since they can open different artifacts on different screens at the same time and use all of the available screen space. The result is depicted in Fig. 11. The labels show the number and rank of the participants and the reference point denotes a maximally challenged hypothetical person. Note that the actual geometric distances of the points are different from what they seem to be in this figure since, for better readability of this illustration, we are using different scales on the three axes.

We used the calculated distances to rank the participants. Then, we propagated the ranks of the participants to the challenges they mentioned and sorted the challenges based on the average of their ranks. Table 3 shows the result.

Discussion Practitioners mostly use multiple screens and multiple windows to work with multiple artifacts at the same time. According to the participants, the most cumbersome task is switching between windows (C2.1), and it gets worse when the tools only support multiple tabs (instead of multiple windows). To make navigation between artifacts easier, some of the practitioners arrange windows side by side. This results in easier switching but raises two other challenges: (i) Each artifact has less dedicated space; therefore, the user has to work in a smaller window (C2.2).

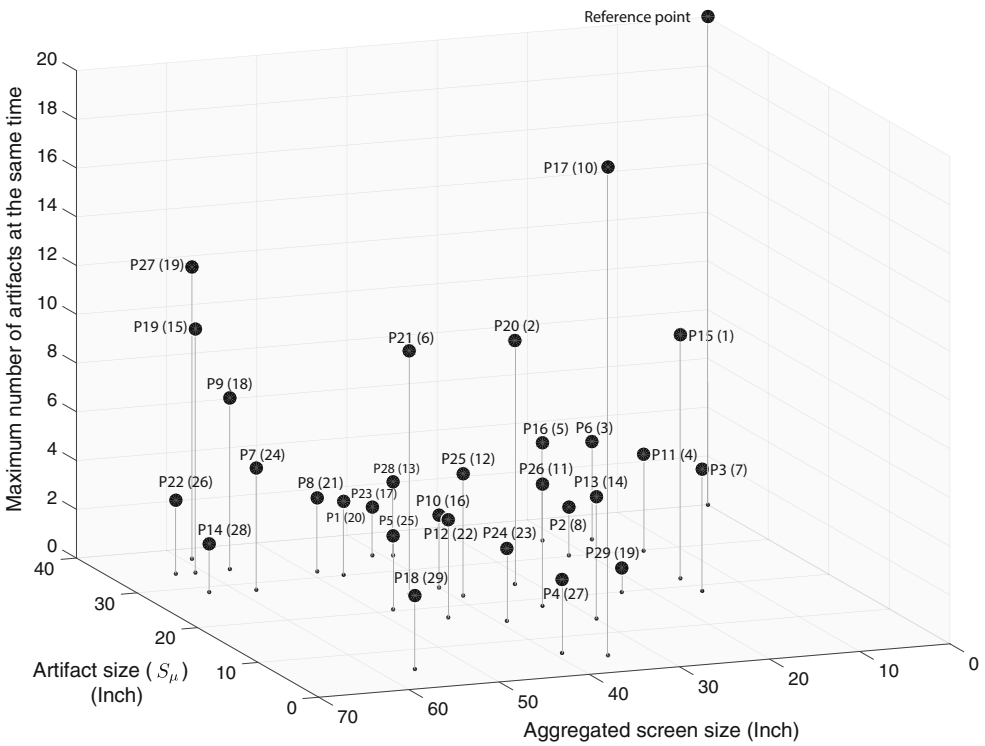

Fig. 11 Ranking of the participants with respect to artifact size, screen size and number of artifacts used. The ranks of the participants are determined by measuring their Euclidean distance from the reference point in the three-dimensional space

Table 3 Challenges of working with multiple artifacts at the same time. Prioritization is explained in the text

ID	Challenge	Priority	#Participants
C2.1	Switching between windows	1	23
C2.2	Working in too small windows	2	12
C2.3	Changing focus	3	8
C2.4	Knowing the relations between artifacts	4	6
C2.5	Finding the right window	5	16
C2.6	Arranging windows	7	4
C2.7	Memorizing	8	10
C2.8	Finding the current focus position	9	2

"[P16:] *Most of the time working with windows side by side is really difficult because there is less space to work*". (ii) The windows need arrangement (C2.6). "[P22:] *I arrange the windows regularly and resize them. When you expand or close a window you have to move the other windows*". When moving back and forth between multiple artifacts, practitioners may forget the exact location where they left an artifact and need some time to find the right location when they return. Therefore, changing focus causes interruption in their work (C2.3). Finding the window that they need in a particular moment is a challenge (C2.5), since windows tend to hide under each other and using keyboard shortcuts to move through windows consecutively takes time and is error-prone. Recent operating systems (e.g., OS X) provide an overview of the open windows, but this becomes increasingly useless when a large number of windows is open. Moreover, bringing up the overview still takes time and does not work with multitab systems. When using multiple windows, there is no information about the relationship between the artifacts inside the open windows (C2.4). Since the space for each artifact is smaller, more information is located outside of the screen. Therefore, the users have to keep more information in their mind (C2.7). The location of the cursor is easily mistaken when multiple windows are open (C2.8). Using more than one screen alleviates some of these challenges to some extent, e.g., switching between windows and memorizing. However, this solution does not resolve those challenges completely and leaves the other ones unaffected. It may also give rise to new challenges such as switching focus between screens and remembering in which screen which artifact resides.

Finding FB3 Storing insufficient artifact relationship information provokes creating larger artifacts and makes searching and understanding artifacts more demanding.

Evidence for FB3 We found that practitioners do not store the relationships between different artifacts in a systematic way, because (i) software tools do not keep relationship information between artifacts (FC7) or, (ii) the artifacts are managed in different tools (FA5), which makes interconnections very difficult. At the same time, we found that most of the participants work on multiple artifacts simultaneously (FA3), which shows that artifacts are indeed interrelated. This motivated us to know what benefits would be gained if a comprehensive interrelationship view of all artifacts were possible. Therefore, we looked for the challenges that practitioners faced due to the lack of relationship information. In particular, we looked for sentences such as: "[P16:] *If I know the relationship between the artifacts of the project, I can imagine a large picture of the artifacts in my mind and I can organize my tasks*" or "[P11:] *When I want to understand an artifact, I spend half of my time on finding the relating artifacts*". We then consolidated the challenges found into six groups (see Table 4). The most frequently mentioned challenge (52%) is that practitioners lose the big picture of the artifacts (C3.5). They do not know the order of artifacts and which artifacts are complementary to each other. Consequently, they are less organized in handling the artifacts. The second most mentioned challenge (48%) is that understanding the artifacts gets more difficult when there is no information available about the related artifacts (C3.4). This challenge is particularly obvious when a new member joins a team. The new member will be disoriented when receiving a stack of artifacts without any relationships. Inconsistency can be caused by unawareness of the artifacts' relationships, as mentioned by 34% of the participants (C3.5). A practitioner can easily miss one of the

Table 4 Challenges of not storing interrelationship information properly. Prioritization is explained in the text

ID	Challenge	Priority	#Participants
C3.1	Forced to create large artifacts	1	2
C3.2	Search for information	2	5
C3.3	Inconsistent change	3	10
C3.4	Difficulty to understand	4	14
C3.5	Loss of the big picture	5	16
C3.6	Reliance on memory	6	2

related artifacts when changing an artifact. 17% of the participants stated that searching for information needs more time when the relationships are not known (C3.2). Two participants (7%) pointed out that they have to use more memory when they do not have a good method to store relationship information (C3.6). Finally, two participants specifically mentioned that not having a decent way of storing relationship data forces them to create very large artifacts to avoid more relationships (C3.1). One of them stated "[P9:] *Generally I do not like to create several artifacts because when I want to change something I should be careful to change all of them. I prefer to have them in one place*".

Discussion As with the previous challenges (FB1 and FB2), we prioritized the challenges we found. For this purpose, we assume that a practitioner with more interconnected artifacts and less effective methods to store relationship information is more affected. From the interviews, we knew the methods of storing interrelationships between artifacts that each participant used. In order to compare the effectiveness of those methods, we included a question in the follow-up survey and asked the survey participants how much relationship information each method captures. The participants had to rate each method on a five-point Likert scale. In addition, they had space to write any method that did not exist in our list. The responses to the open part of the question were methods that basically belonged to one of the items of our list, e.g., "traceability matrix" which is an extra artifact and "tagging" which is a tool feature. Table 5 shows the effectiveness of storing interrelationship information from the viewpoint of the survey participants after quantifying.

In Fig. 12, we defined a two-dimensional space based on two parameters: (i) the effectiveness of storing relationship information and (ii) the number of artifacts used at the same time (discussed in FA3). To calculate a value that shows the effectiveness of the methods that each participant used for storing relationship information, we first extracted the mean value for effectiveness of each method from the survey results. Then, using these extracted values, we calculated mean values for the participants based on the methods they mentioned in the interview. In Fig. 12, each point represents

Table 5 Effectiveness of storing interrelationship information

Storing interrelationship method	Mean of effectiveness
Software tool	4.0645
An extra artifact	3.5806
References to other artifacts	2.9354
Folder structure	2.5161
File name convention	2.3225
Memorizing	1.9032

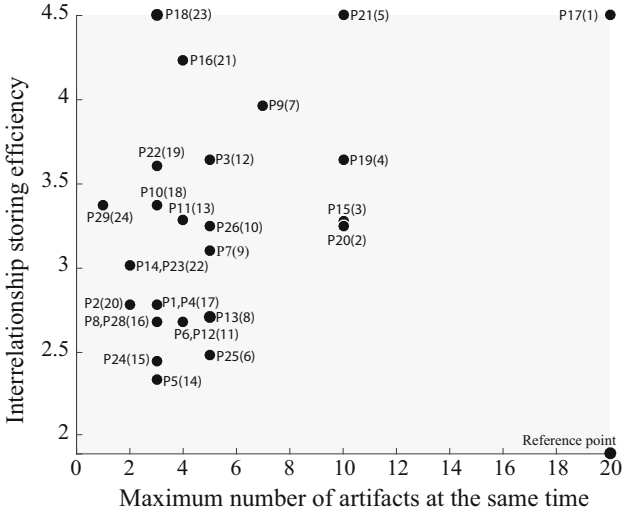

Fig. 12 Ranking of the participants with respect to artifacts used at the same time and the effectiveness of the method they use. The ranks of the participants are determined by measuring their Euclidean distance from the reference point

a participant. The reference point represents a hypothetical person who has the highest number of connected artifacts (20) and the least effectiveness of storing relationship information encountered in this study (1.9).

The Euclidean distance between each participant and the reference point shows how much that participant is affected by the challenges caused by poorly stored artifact relationship information. Note that the actual distances of the points are different from what they seem to be in this figure since, for better readability of the diagram, we are using different scales on the two axes. We ranked the participants based on their distances and used the ranking to prioritize the challenges. The labels show the number and rank of the participants. At the top of the prioritized list of challenges, we have a challenge which is mentioned only by two participants. This shows that the mentioned challenge is not of less importance than the others. In fact, this type of challenges that are important and mentioned by few participants are more difficult to track down. In our case, having no good means of storing relationship information unconsciously forces the practitioners toward creating larger artifacts (C3.1). Large artifacts are difficult to understand and give rise to other challenges we discussed in this paper (FB1).

4.3 Dealing with the challenges (RQ3)

We asked the participants how they deal with the challenges of handling artifacts. In particular, we were interested in knowing what methods they use to overcome the challenges mentioned above, what features their tools provide, and what they do when the tools do not support them sufficiently.

Finding FC1 Practitioners use their memory extensively.

Evidence for FC1 One of the main challenges in working with large artifacts, according to FB1 and FB2, is "relying on memory". To know how many of the participants use their memory intentionally, we asked them directly whether they memorize any part of an artifact to use it elsewhere or not. 69% of the participants answered affirmatively. We asked the participants who responded "no" how they handle the situations where they need a piece of information from another artifact or another part of the current artifact which is not on the screen at the moment. We found that 21% of the participants use copy and paste functions instead of keeping information in memory. They also mentioned that this method is not applicable to graphical information easily and they sometimes have to take screenshots. "[P19:] *I don't memorize. Instead, I use copy-paste. If it is a diagram, I would make a screenshot of it*". Finally, only 10% of the participants answered "no" decisively.

We asked the participants if better visual memory positively influences their performance. 82% of the participants admitted that better visual memory affects their performance positively in working with graphical artifacts.

Discussion The answers to these two questions prove that practitioners rely on their memory extensively when working with artifacts and show that they compensate their inadequate memory power by using copy-pasting and taking screenshots, which is error-prone and time-consuming in turn. Larger screens and additional screens are helpful to reduce the amount of information that practitioners keep in their memory. However, the screen size is limited and in many situations one artifact is not possible to be opened multiple times. In addition, working with an artifact spanned over multiple screens is challenging.

Finding FC2 Traditional zooming and scrolling are the dominating techniques for handling large artifacts.

Evidence for FC2 As stated in FA1 and FA2, practitioners often work with artifacts that are larger than their screens. Many artifacts even do not fit on the largest screens reported in this study (FA2), i.e., practitioners cannot view the entire artifact on the screen after zooming out in a readable size. However, when we asked them how they handle such artifacts and what features tools provide for this purpose, we found that they mostly use simple traditional methods such as scrolling and zooming.

Cockburn [5] categorized visualization techniques that help handling larger-than-screen artifacts into four classes: zooming, overview + detail, focus + context and cue-based techniques. We asked our participants how they handle large artifacts to know the techniques implemented in commercial tools. We found that traditional zooming and scrolling are the most basic techniques used for this purpose. In addition to zooming and scrolling, only three participants use tools that provide an overview + detail feature. The applications they use show an overview of the artifact in a small window and they can navigate inside the artifact by using this small overview. None of the interviewees have any focus + context or cue-based techniques available. Obviously, the features that exist but are not known by the participants are not counted in this report. Three interviewees explained that they avoid having large artifacts by defining different layers of abstraction. The result is visualized in Fig. 13.

Discussion By maintaining multiple layers of abstraction, some of the participants could manage to have smaller artifacts at the cost of increasing the number of artifacts and having redundant data in multiple artifacts. Managing a larger number of artifacts with redundant data needs additional effort. Working with multiple artifacts gives rise to other issues that we discussed in FB2. In this regard, participant 18 describes his needs as "In my tool, different diagrams which show different layers of abstraction cannot be interconnected. What I really like is to start on a high level and go to a really detailed level, and get to the other diagrams that show the layers beneath. I do not know any software that has this kind of zooming".

Finding FC3 Almost all practitioners need to have an overview of the artifacts.

Evidence for FC3 The participants told us that they need to have a rough image of the entire artifact when working on it. This image makes them aware of the size, complexity, different sections, position of the important elements, and the elements' types of an artifact. 65% of the participants keep an overview of the artifact in their mind and 32% of the participants employ other techniques to maintain an overview of the artifact (see Table 6). Only

Fig. 13 The percentages of the methods that participants used to handle artifacts that are larger than the screen

Table 6 Alternative techniques to maintain overview

ID	Alternative technique to maintain overview
P2	Zooming out
P3, P16	Printing and hanging the artifact on the wall
P12	Taking notes
P15	Opening the artifact twice
P17, P18, P20	Creating a higher abstraction level of the artifact
P23	Using the overview provided by the application

one participant mentioned that he does not need to have an overview.

Discussion The list of techniques given in Table 6 reveals that participants primarily use simple techniques (e.g., zooming out or opening an artifact twice) or improvised workarounds (e.g., printing or taking notes). All of these methods are not equally appropriate for all types of artifacts, e.g., only graphical artifacts can be printed and hung on the wall. The only sophisticated and systematic technique employed is creating a higher abstraction level. However, this is complicated and time-consuming as the participants have to do this manually.

Finding FC4 Non-software approaches are mostly used for handling the challenges of working with multiple artifacts at the same time.

Evidence for FC4 To handle the challenges related to the number of artifacts used simultaneously, 79% of the participants use multiple screens (two or three screens) and 24% of them print some of their artifacts in addition to having multiple screens (Fig. 14).

Discussion Using multiple screens and printing increases the number of artifacts that practitioners can view simultaneously without any switching. The fact that the majority of practitioners uses multiple screens demonstrates their need for concurrently working with more than one artifact. However, both multiple screens and printing have drawbacks and limitations. The main drawback is that both exacerbate the challenge of repetitive change of focus, which is one of the main challenges we found (FB2). Moreover, the number of concurrently usable screens and printouts is limited in three dimensions: (i) cost, (ii) available space to place screens and printouts in a work environment, and (iii) at some point searching for the needed information in a multi-screen and multi-printout environment becomes as cumbersome and difficult as keeping the same information in multiple windows on a single screen.

Finding FC5 Paper is used by all and whiteboards are used by two-thirds of the participants for creating artifacts.

Evidence for FC5 When we asked the participants whether they used any non-software ways to create, understand or manage their artifacts, we most often received "pen and paper" or "whiteboard" as an answer. Among 29 participants, 28 participants use pen and paper and 22 participants use a whiteboard.

Discussion As discussed previously (FA5), using multiple tools to deal with artifacts has drawbacks. Software development practitioners tend to use as few tools as possible. When they decide to use a new tool (software or non-software) in addition to what they already use, it means that their current tools do not satisfy their needs completely. To find the shortcomings of the tools we asked about their reasons to use pen and paper or whiteboards, which will be discussed in FC6.

Fig. 14 The percentages of the methods that participants used to handle multiple artifacts at the same time

Finding FC6 The reasons for using paper and whiteboards for artifact creation include seeking more speed, flexibility and space to work.

Evidence for FC6 As the reason for why non-software tools (pen and paper or whiteboards) are used to create artifacts, the most mentioned one was "the ease of use". We did not count this reason as a finding because being easy depends on many other factors. Instead, we looked deeper in the content of the interviews to extract the real reasons. For instance "[P22:] *It is easier to brainstorm on a whiteboard. Everyone can see it and you can erase things very quickly*" or "[P27:] *I can draw anything that is in my mind and you can find a piece of paper anywhere*". The participants mentioned that drawing on paper is easier because of the following six reasons. (i) They are not limited to the rules of a tool and they can draw whatever they want. This means that drawing on paper and whiteboards is more flexible. (ii) They have more space, especially on a whiteboard. (iii) They easily have the big picture of the artifact available. (iv) They can share it with others with no effort. (v) Paper is portable and available everywhere. (vi) They can draw faster. "Being fast" is partially dependent on other factors just like "being easy", but is partially a genuine feature of drawing on paper and whiteboards. That means, drawing on paper and whiteboards can be faster due to reasons (i)–(v) given above. Another reason could be that creating the details of a diagram (such as boxes with text) may be faster on paper or whiteboards than with a graphic tool.

Discussion Drawing on paper and whiteboards is not necessarily a problem; it may even provide practitioners with benefits. However, when drawings on whiteboards and paper go beyond throw-away sketches, they need to be digitized at some point of time. The time and effort spent for digitizing artifacts can be saved if they are drawn in a digital tool in the first place. This can be possible if software tools provide the practitioners with the same level of ease as paper and whiteboards. In addition to being time-consuming, the digitization process is also error-prone since practitioners may make mistakes when they want to recreate a diagram in a software tool or they may misread what is drawn on the whiteboards or paper [61].

Finding FC7 To store artifact interrelationship information, practitioners extensively use inefficient, time-consuming and error-prone methods.

Table 7 Distribution of methods of storing artifact relationship information

Store relationship information method	Percentage (%)
Folder structure	62
Software tool	27
File name convention	27
Reference to other artifacts	27
An extra artifact	20
Memorizing	17

Evidence for FC7 Table 7 shows how our participants reported to keep the relationship information of the artifacts. Only 27% of them use the ability of their software tools to keep this relationship information, and only 10% of the participants do not use any other way concurrently, i.e., they use only their software tool. According to the gathered data, practitioners mostly use folder structures to show which artifacts are related to each other (62%). In addition, 27% of the participants mentioned that they have file naming conventions in their company to show the relationships between files, e.g., by starting the name of related files with similar prefixes. 27% mentioned that they reference artifacts explicitly in other artifacts. 20% of the participants create an extra artifact that contains the information about the relationships between artifacts. Finally, more than 17% of the participants keep this information in their mind.

In the follow-up survey, we mentioned the classified methods as shown in Table 7 and asked the practitioners, as an open question, if they know any other method of storing relationship information. The majority agreed that the list is complete, while some mentioned the following two methods: automatic traceability techniques based on the similarity between files, and tagging artifacts. The former is a method being researched vastly [59] and may be used in industry in the future, and the latter is a widely used method in different types of applications, e.g., personal task managers and social networks. Since this method is mentioned only once in the follow-up survey we will not include it in our analysis. In Table 7, the sum of the percentages is more than 100% since many of the practitioners use more than one notation concurrently.

Discussion Folder structure and file name conventions can keep a limited amount of information about the artifacts and need all the members of the team to pay attention to maintain them. When referencing artifacts in other artifacts, it is not possible to have an overview of the relationships and searching is not convenient. Furthermore, this method needs maintenance whenever one of the artifacts is changed. Creating an extra artifact to keep the relationships between artifacts provides the practitioners with an easily searchable overview of the relationships. However, this needs extra effort to create and maintain. Otherwise, practitioners would always end up with outdated and useless data. The most inefficient and error-prone approach is keeping the information in mind. The practitioners may forget and make mistakes. A new member of the team would have no understanding of the relationships in such an environment. Participant 26 brought this to the point: "I know the connections by heart, but when someone is new it is very confusing for him".

In addition, in the follow-up survey, we asked the participants to rate the amount of effort needed to store the relationship data in each method. As Fig. 15 shows, the amount of effort needed to make software tools keep the relations between artifacts is much higher than that required for folder structure and file name conventions. This amount is very close to the amount of effort needed for referencing inside artifacts and creating an extra artifact. This assessment justifies the practitioners' decision about which method they use. They use folder structures and file name conventions because they are easier, and use referencing inside artifacts and creating an extra artifact since they are not much more difficult. In the follow-up survey, we asked the participants to rate their overall satisfaction with the available methods to store relationships between artifacts on a five-point Likert scale. The result is presented in Fig. 16. The majority of

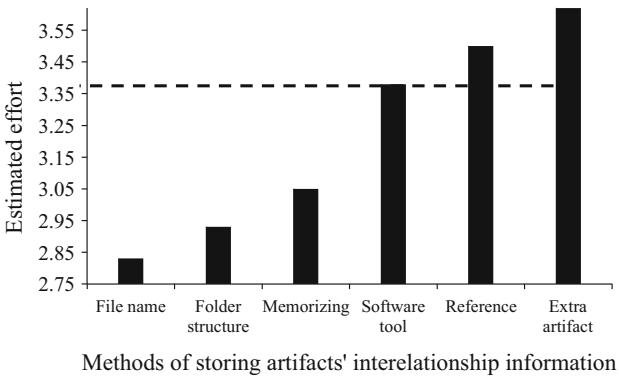

Fig. 15 The effort needed to setup and maintain different methods of storing interrelationship information. The corresponding value for "Software tool" is emphasized by a horizontal dotted line

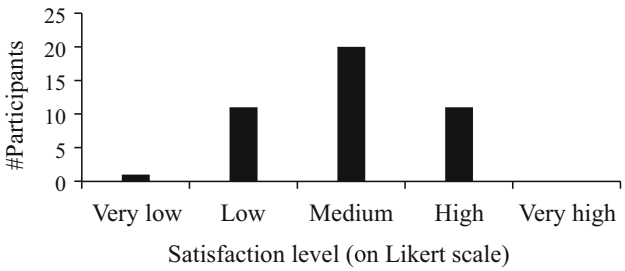

Fig. 16 Participants' overall satisfaction with the available methods to store artifact interrelationship information on a five-point Likert scale

participants are not satisfied, with a satisfaction level in the range of "Very low" to "Medium".

This shows that a decent method for storing relationship information is needed, but currently not available.

5 Consolidation of findings

So far, we presented the answers that we found for the first three research questions (RQ1–RQ3) in the form of findings. A summary of these findings is given in Table 8. Before answering the fourth research question in the next section, in this section, we consolidate our findings into a single comprehensive view and show how the challenges can impact usability factors.

A tool requires both sufficient diversity of features and usability to succeed. In other words, a very feature-rich software product which is not usable cannot deliver its potential benefits to the users. For example, Google Wave was a feature-rich web application which was discontinued after 2 years due to usage complexity [57]. While a large body of research studied ways to provide richer application tools and other researchers assessed the usability of the tools, we decided to look at the problem from a different viewpoint and find how the interaction challenges that practitioners experience in working with artifacts affect usability factors of tools.

Before connecting our findings to usability factors, we need to look at the definition of usability. The ISO 9241 standard Part 11—Guidance on Usability (ISO, 1997) defines usability as follows [25]: "Usability is the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use." According to this definition, there are three factors which determine the usability of a tool: effectiveness, efficiency, and satisfaction [18]. *Effectiveness* is the precision and completeness with which users achieve their goals. Effectiveness can be assessed by the quality of the solution and the error rates. *Efficiency* is related to the amount of effort invested to achieve goals. It can be assessed by task completion time and learning time. *Satisfaction* is the users' comfort with which the goals are achieved. Users' satisfaction can be measured by attitude rating scales.

In Fig. 18, we consolidate all our findings into a single view and show how they influence the three usability factors effectiveness, efficiency, and satisfaction. In order to create an expressive view, we used the pattern described in Fig. 17.

Table 8 Key findings

RQ1: Properties of artifacts, screens, and tools	
FA1	Only about one-third of the graphical artifacts used by the interviewed practitioners fit on their screens
FA2	About forty percent of the graphical artifacts do not fit on the largest screen reported in this study
FA3	More than half of the interviewed practitioners use four or more artifacts at the same time
FA4	More than two-thirds of the interviewed participants use customized notations for their artifacts
FA5	On average, practitioners use more than five applications concurrently to create and manage their artifacts
FA6	Although almost all participants work on their artifacts collaboratively, less than one-third of the collaboration is done with dedicated software development tools
RQ2: Challenges practitioners encounter when working with artifacts	
FB1	"Relying on memory", "Searching for information", and "Maintaining the overview" are the most important challenges in handling large artifacts
FB2	"Switching between windows" and "Working in too small windows" are the most important challenges when working with multiple artifacts
FB3	Storing insufficient artifact relationship information provokes creating larger artifacts and makes searching and understanding artifacts more demanding
RQ3: Dealing with the challenges	
FC1	Practitioners use their memory extensively
FC2	Traditional zooming and scrolling are the dominating techniques for handling large artifacts
FC3	Almost all practitioners need to have an overview of the artifacts
FC4	Non-software approaches are mostly used for handling the challenges of working with multiple artifacts at the same time
FC5	Paper is used by all and whiteboards are used by two-thirds of the participants for creating artifacts
FC6	The reasons for using papers and whiteboards for artifact creation include seeking more speed, flexibility and space to work
FC7	To store artifact interrelationship information, practitioners extensively use inefficient, time-consuming and error-prone methods

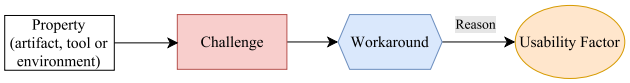

Fig. 17 The pattern used to create the chains that are building blocks of Fig. 18

The consolidated view in Fig. 18 consists of several chains that follow this pattern. Each chain starts with a property (▭) of artifacts, tools or environments. Each property is connected to one or more challenges. Each challenge (▭) is connected to one or more workarounds (⬭) that practitioners employ to deal with that challenge. The workarounds are connected to the corresponding usability factors (⬭) that they negatively influence. Labels on influence arrows describe the rationale for the influence.

Other influence relations, which do not follow the pattern given in Fig. 17, are depicted with dotted arrows (⇢). They may show a workaround influencing another workaround (e.g., using multiple tools causes opening more windows), a workaround affecting a property (e.g., using multiple tools makes keeping interrelationship information more difficult), or a property that intensifies another property (e.g., having larger artifacts results in partially visible artifacts).

In the subsequent paragraphs, we provide a detailed description of the chains from the six properties shown in Fig. 18 to the three usability factors, and show how our findings (cf. Table 8) relate to these chains.

P_1–Collaborative work Practitioners work on their artifacts collaboratively. The first step to collaborate on an artifact is sharing the artifact. We found that the practitioners use methods such as whiteboards, paper, file sharing services and general collaboration tools since their dedicated requirements engineering or software development tools do not support all the collaboration features they require (*FA6*). Using paper and whiteboards affects the efficiency due to the time needed for digitization, and affects effectiveness due to the errors that occur during digitization (*FC6*). Sharing files is a quick workaround, but needs synchronization between collaborators. There is a possibility of changing something that should not be changed or overwriting others' work. Both decrease the effectiveness of this workaround. Some practitioners share their monitors to collaborate on an artifact (*FA6*). In this setting, only the sharing collaborator can change the artifact, while all others can only read it, which is not satisfying. In addition, only synchronous collaboration is supported and the collaborators have to dedicate their whole attention to the collaboration, which is not efficient. Another workaround for this challenge uses a specific collaboration tool (e.g., Google Docs) (*FA6*). General collaboration tools decrease the probability of conflicting with each other. However, these tools support only a limited number of file types. Therefore, the owner of an artifact should transfer it to the collaboration tool and return it to the original tool when the collaboration is done. This limitation makes this workaround inefficient. Generally, using a dedicated collaboration tool increases the number of tools that are used by the practitioners concurrently, thus making them open more windows at the same time. This workaround is discussed in P_3.

P_2–Custom notation We understood from the interviews that practitioners like to draw by hand (*FC5*). One of the reasons for this is that they do not always want to comply with the notation rules (*FA4*), especially at the beginning, when an artifact is more subject to change. This is challenging since most of the tools want their users to obey the syntax of the languages they support, while the users need flexibility. A workaround for this challenge is using paper and whiteboards (*FC6*), which affects efficiency and effectiveness, as mentioned in P_1. Another workaround for this challenge is using the most suitable features of different tools (*FA5*). For example, a practitioner may create one type of artifact in a tool and another type of artifact in a different tool. When using multiple tools, practitioners have to open multiple windows simultaneously (*FC4*). We describe this workaround in P_3. Using multiple tools also intensifies the property of low interrelationship information of the artifacts (P_4) (*FC7*).

P_3–Related artifacts The artifacts are related for different reasons. Due to these relationships, when a practitioner is performing a task on an artifact, specific information from other artifacts is sometimes required and the practitioners search for that information. Moreover, practitioners sometimes need to work on multiple artifacts at the same time (*FA3*). To overcome these two challenges, practitioners open multiple artifacts in multiple windows or tabs simultaneously. When switching between these windows, the practitioners lose their focus and may commit mistakes, which affects effectiveness negatively (*FB2*). In addition, they spend time for managing the open windows, i.e., they arrange, hide and unhide windows according to their current task. This overhead makes their work less efficient. Also, during windows management and switching between windows, practitioners may have difficulties in finding the right window among the open windows, or they may switch to a wrong window and change something before they find out that they are in a wrong place (*FB2*). These annoying occurrences are unsatisfactory. Opening multiple windows often causes the windows to get smaller. Smaller windows intensify the property of partially visible artifacts (P_6). Scrolling is used as a workaround for searching information in partially visible artifacts, therefore we will address it in P_6. In addition to the challenges mentioned, this property (related artifacts) makes storing information about the artifact interrelationships (P_4) more challenging (*FC7*).

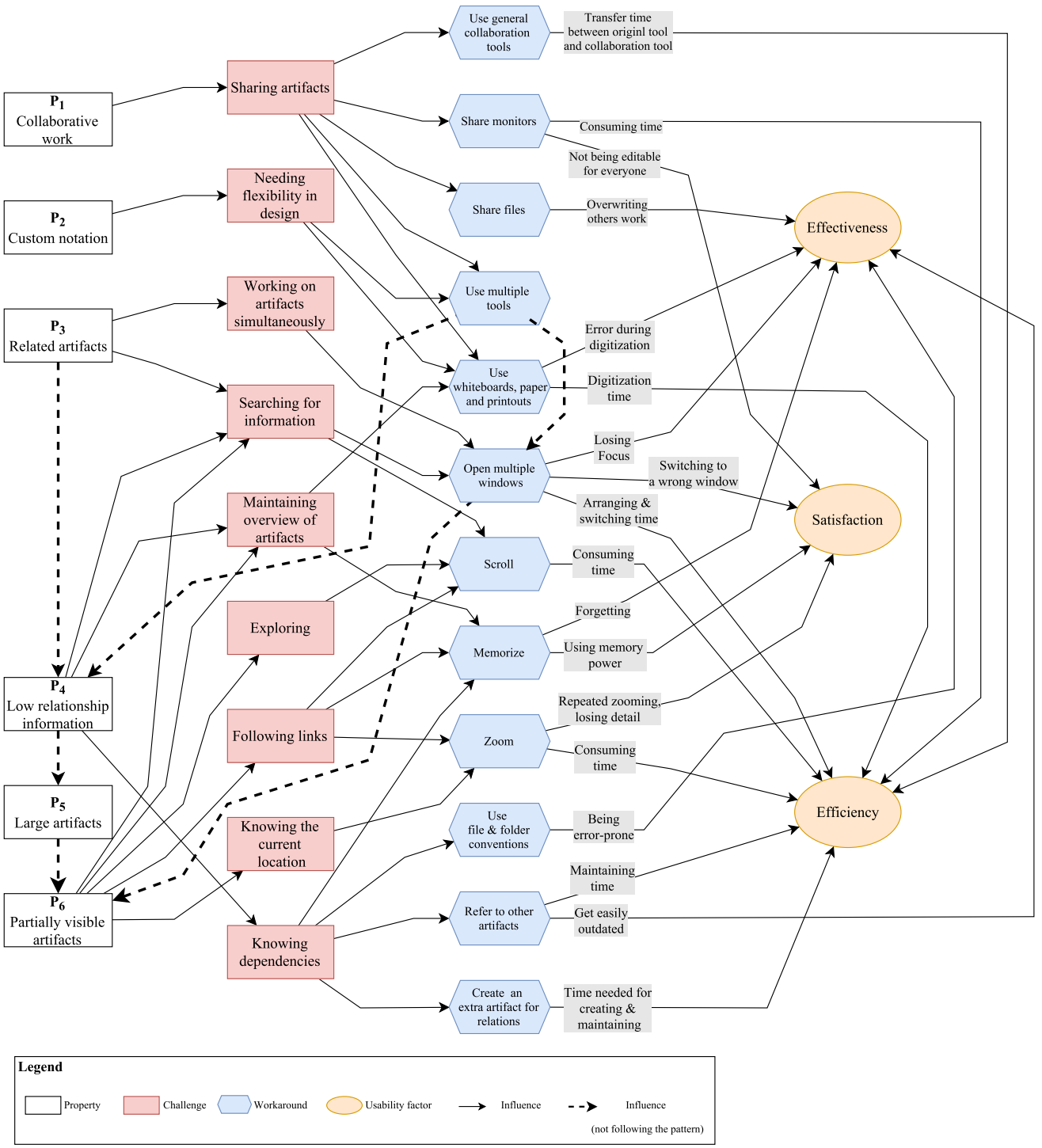

Fig. 18 A consolidated view of all the findings of this paper. The diagram consists of several chains starting from a property and ends with a usability factor. The pattern of the chains is depicted in Fig. 17. A full description of the chains and their relations to the findings is available in Sect. 5

P_4–Low interrelationship information Absence of interrelationship information between artifacts causes three challenges. One challenge is searching and finding information (discussed in P_3 and P_6). Another challenge in such a situation is that the practitioners cannot easily create and maintain a mental image of the artifacts' relationships (*FB3*). As a workaround, they rely on their memory (*FC1*) or print their artifacts on paper and hang them

on the wall to have an overview always available (*FC3*). Relying on memory affects effectiveness since the practitioners may forget some parts and affects their satisfaction since they need more effort to accomplish their tasks. Another challenge caused by low interrelationship information between artifacts is knowing about the dependencies between artifacts. Knowing these dependencies is crucial for understanding the artifacts and for keeping them consistent (*FB3*). In addition to relying on their memory, practitioners use other workarounds. They use file and folder naming conventions to show the related artifacts, which keeps a limited amount of information and needs to maintain the attention of all team members. Since the system does not enforce these conventions, making a mistake can happen frequently when employing this workaround (*FC7*), making it ineffective. Some of the practitioners place references to other artifacts into an artifact. Since updating all these references is difficult in the case of changes, there are always some outdated references in the artifacts which make this method less effective. It is not efficient due to the time needed for maintenance. The last method for storing interrelationship information creates an extra artifact that shows the relations. Although this method is effective, its high cost to create and maintain such extra artifacts makes it less efficient (*FC7*). In addition to the challenges that unavailability of the interrelationships causes, we found that it discourages creating related artifacts and encourages creating large artifacts instead (*FB3*).

P_5–Large artifacts We found that a considerable number of artifacts are larger than the screens on which they are displayed for viewing and editing (*FA1, FA2*). Being large does not make working with an artifact challenging per se. However, when a large artifact is viewed on a limited-size screen, a part of the artifact always remains invisible. This problem is described in P_6.

P_6–Partially visible artifacts As discussed earlier in P_3 and P_5, opening multiple windows and having large artifacts result in views where only a part of an artifact is visible. This causes several challenges. It makes searching for information and maintaining an overview of the artifacts more challenging. In addition to opening multiple windows (discussed in P_3), practitioners scroll long distances to reach the information they need, which is time-consuming and affects efficiency negatively. Also, practitioners need to explore an artifact in order to understand it (*FC2*). Exploring becomes more challenging when a large part of an artifact is invisible (*FB1*) and practitioners scroll extensively to explore the artifact (*FC2*). Scrolling negatively affects efficiency due to the time it consumes. Partial visibility also makes it more challenging to follow links, because most of the destinations of the links will be outside of the screen and practitioners have to scroll or go to another window to find out where the links end (*FB1*). This is cumbersome and time-consuming, thus negatively affecting efficiency. In addition, the practitioners use the zooming workaround to follow links (*FB1*). They zoom out to view the links and then zoom in afterward to continue their work. In addition to being time-consuming, which affects efficiency negatively, repeated zooming actions and losing detail when the artifact is scaled down are annoying and make users feel unsatisfied. Practitioners also use zooming as a workaround for the challenge of finding their current location in an artifact.

6 Existing work addressing the challenges

In this section, we survey existing work that addresses the challenges we found in our study (see Sect. 4.2). Our purpose is to explore to which extent there exist solutions to these challenges, thus providing a preliminary answer to our research question RQ4 (RQ4: Which existing solutions address the challenges of working with RE artifacts?). For definitely answering RQ4, a systematic literature review would be required, which is beyond the scope of this paper. We present existing solutions in four research directions: *maintaining artifact interrelationships*, *flexibility in working with artifacts*, *collaboration*, and *navigation inside and between artifacts*.

6.1 Maintaining artifact interrelationships

Practitioners use multiple tools and each tool provides a set of functionalities and produces a set of artifacts. The artifacts are related to each other like a network. Practitioners need interrelationship information to perform different tasks. They spend a considerable amount of time, cognitive power and effort to compensate the lack of such information. In addition, considering the challenges of handling related artifacts and storing interrelationships between artifacts, a centralized view showing the relationships is necessary.

Traceability is an extensive concept in software engineering. One of its aspects is defined as the degree to which a relationship can be established between two or more products of the development process [24]. Among other benefits, having a traceable software development process enables the team to know the relationship between artifacts. Numerous studies are done in the field of traceability to maintain the traceability links from the beginning or to retrieve them when they are not kept properly. A survey in this field has been done by Winkler et al. [59]. The question is if these tools are used in the software development industry. Müller et al. [39] found in an empirical study that the links between artifacts are often missing. They only mentioned face-to-face communication and wiki pages as

the workarounds that practitioners use to find the relations between the artifacts. In another study, de Souza et al. [55] found that the development teams that they studied relied on emails and instant messaging to find the change impacts. Many researchers study systematic ways of retrieving interrelationship information in the form of traceability links [59].

Software development tools can save practitioners' effort and facilitate the traceability link identification by storing the relationship information as the artifacts are created and evolved. For example, Codebook [1], which is a framework to connect practitioners and artifacts in a single directed graph, makes it possible to conveniently find the interrelationship information of artifacts. It keeps the relationship between artifacts and developers in a social network so that developers know whom to ask a question about the artifact they are working on.

Using one software development environment is an approach to have all artifacts in one place. However, developing software tools that address all needs related to software development artifacts is very difficult. One way to provide such a view is to have a centralized repository of artifacts. For example, IBM rational DOORS is a requirements management system that imports artifacts of various formats. It allows the requirements engineers to create and maintain relations between parts of different artifacts. However, working on artifacts in their native applications and synchronizing them with DOORS repository requires effort. Another way to have a central management system while allowing requirements engineers to use their desired editing application is to create different small tools, instead of one large tool, such that they can be plugged into other tools. This is already done in a few tools for selected features, e.g., Confluence can plug in the drawing tool Gliffy. In this case, there is a central management tool and, at the same time, users can search for the pluggable tools that match their needs. Another way to achieve a centralized view would be to define a standard for artifact definition and let different tools store their artifacts in a shared repository. For example, XMI is created for UML diagrams so that different tools can store their artifacts in a common format. A common repository can be created based on a common standard way of storing artifacts.

6.2 Flexibility in working with artifacts

One of the reasons why practitioners need to use whiteboards and paper to design their artifacts is that they cannot draw exactly what they want in the tools and are limited to the templates, styles, and notations that tools provide. Furthermore, practitioners like the convenience of drawing by hand. To achieve more flexibility, tool developers should think about methods to bring the convenience of hand drawing to their tools without turning them into mere painting tools. We are aware of two approaches in this direction. *FlexiSketch* [61, 62] looks like a free-hand drawing tool at the first glance, but it seamlessly transforms what the user draws into a diagram consisting of meaningful, related components. In addition, during this process, it lets the user define a custom notation. *Calico* [37] is a sketch-based design tool for touchscreen devices. Its focus is supporting the early stages of software design. Using Calico, practitioners can have an experience similar to paper and whiteboards when drawing and improving their designs on electronic whiteboards or tablets, while Calico enhances their experience seamlessly. In 2010 and 2011, two workshops on flexible modeling tools [42, 43] were held to investigate the reasons why practitioners prefer to work with whiteboards, paper, and other informal methods at early stages of software modeling and development, instead of using specific modeling tools.

6.3 Collaboration

The development of a complex software system is often a highly collaborative process, where ideas are presented and discussed by multiple stakeholders [13]. The requirements engineers and software designers collaborate to create an artifact or to describe it to each other. In a distributed collaboration, the tools need to have the same features as non-collaborative versions with the additional ability to let multiple users work on a single artifact, while in a collocated setting a larger screen is needed so that all the users can work on it simultaneously [14]. Various tools have been proposed for distributed software design and development. In this regard, a systematic mapping review was done by Portillo-Rodríguez et al. [47]. IBM Jazz is an extensive collaboration platform that covers various areas of a software engineer. Rational Requirements Composer is a tool based on this platform and lets requirements engineering collaboratively work on requirements artifacts. A hybrid approach to solve the collaboration problem is proposed by Wüest et al. [63] in which each collaborator has his/her own device (tablet) while a large screen and a large electronic whiteboard is their common point of reference.

General collaboration tools allow users to edit even the same sentence simultaneously without interrupting each others' work. For this, Google Docs is an example. Software development tools should incorporate these successful collaboration practices so that practitioners can easily collaborate without leaving their software development environment. Such features encourage collaboration, improve awareness and save time. There are some software development tools that allow collaborative work, such as letting users have shared Scrum boards (e.g., Trello) or edit their wiki pages collaboratively (e.g., Confluence).

However, they do not support all types of artifacts and all types of collaboration.

6.4 Navigation inside and between artifacts

Practitioners spend a considerable amount of time using traditional navigation mechanisms such as scrolling, zooming, and opening multiple windows. They make mistakes and perform unsatisfactory actions while using these mechanisms. All of these affect their productivity negatively. In addition, screen space for displaying information has always been limited. Therefore, the way users interact with the information displayed on the screen has been the focus of a large number of studies. These studies aimed especially at optimizing the presentation of information. Two different approaches can be taken for this purpose: increasing the screen space or utilizing the available screen space more efficiently. In the first approach, using large screens or arrays of multiple screens is investigated. Czerwinski et al. [9] conducted an empirical study to examine the productivity benefits of larger display screens and found a significant performance advantage. Lischke et al. [35] used multiple monitors to have a wall-size screen in an empirical study and measured the task completion time in different settings. They reported that the optimal monitor number is three.

In the second approach, techniques such as zooming, overview + detail, focus + context and cue-based methods are employed to display as much useful information as possible on the screen [5]. Lam et al. [31] analyzed 22 studies that implemented such techniques to extract design guidelines indicating when and how each of these techniques should be used. Generally, tools should use the space of the screen more intelligently, show the needed information only, gather information from multiple sources into a single view, e.g., by employing semantic zooming and different levels of detail [26]. Additionally, tools can reduce the amount of information that practitioners keep in their mind about what exists outside of the screen, e.g., by providing such information on the border of the screen [17].

In this direction, initial steps are taken in some tools. However, what is actually implemented remains far behind the state of research. Some tools show an overview of the artifact in a small window somewhere on the screen which helps the user know where the working region is located in the artifact and lets the user navigate faster. Although this is an old way of dealing with complexity, few tools have incorporated such a feature. As another example, the Fisheye concept [19] has become a standard for source code version control. In such tools, the changed line is highlighted and is shown together with a few lines before and after it. The rest of the source code is hidden and represented by a symbol such as three dots. In a conceptual solution, FlexiView [22] has extended the Fisheye concept to other aspects of artifact navigation to accommodate only the information on the screen that is required for the current task. These tools reduce the zooming and scrolling actions and show information from different sources in a single view to decrease the number of concurrently open windows.

However, both of these approaches give rise to other challenges such as arranging the windows and tracking the mouse pointer [51]. Therefore, for improving the performance of existing user interfaces, it is not sufficient to just increase the screen size or employ a smart visualization mechanism. Instead, the design of user interfaces needs adaptation, which requires understanding the new challenges. Furthermore, the findings of these researchers depend on the information type (e.g., graph or 3D model), the interaction type (e.g., comprehension or manipulation) and the users. Consequently, the provided guidelines need to be tailored to requirements engineering.

Requirements visualization is another broad area of research that investigates how graphical models of requirements should be created [6]. However, the research in this area does not address how the created graphical models should be presented to users. For example, Cleland-Huang et al. [4] proposed visualization techniques such as hierarchy structures to enhance the understandability of artifacts in automatic tracing tools. Reddivari et al. [48] designed a tool to support the exploration of requirements via quantitative visualizations. The true benefits of these tools will not be realized unless the artifacts are presented to users in the most effective form. For instance, Reinhard et al. [49, 50] developed a custom-made presentation technique to fully exploit the potential of the requirements modeling language ADORA.

In addition to enhancing the understandability of the artifacts, software engineering tools can provide cognitive support [45]. If a complex diagram is not presented hierarchically, viewers have to derive the hierarchy in their mind [30]. Cornelissen et al. [7] established a set of metrics for scenario diagrams to recommend a number of abstractions that should be used to have the desired amount of detail. Bennett et al. [2] reported the usefulness of their interaction features for sequence diagram navigation. Presenting information in a clear pattern helps to remember the relationships [30]. In a data-intensive field like requirements engineering, offloading some of the cognitive load is a requirement for any tool which supports viewing and editing of artifacts.

7 Related work

In the previous section, we discussed a number of related studies that addressed the same challenges we found in our study. In this section, we discuss related works that study

RE practitioners, how they work with RE artifacts and how they use tools that support them.

Software engineering heavily involves people. In order to understand software engineering and enhance different aspects of it such as working with RE artifacts, the first step for researchers is to study people empirically. In an empirical study, based on the selected research method, a suitable data collection technique should be chosen as well [15]. Lethbridge et al. [32] compiled a list of various techniques of acquiring information about people and how they work in software engineering environments. They have cited successful prior experiences of such studies and applied their findings to improve software engineering. For example, as data collection techniques, they have used interviews and surveys in another work [33] to study how software engineers use documentation. They reported that documentation is most of the time outdated and documentation is often poorly written. However, their study did not seek for the reasons.

Empirical studies in RE investigate various aspects of RE such as the effects of improved RE over an entire project lifecycle [10], user involvement in software engineering [44], and effectiveness of elicitation techniques [11]. There are few empirical studies of requirements artifacts and how they are used by practitioners. Liskin [36] has interviewed practitioners to understand how they manage to work with multiple artifacts and how they link their artifacts together. Winkler [58] has performed a survey to find which artifacts are created in the requirements phase of software engineering and how information flows between related artifacts. The last two mentioned studies consider the artifacts and their usage without considering the tools that make using artifacts possible. In contrast, we look for the interaction challenges of working with artifacts using tools.

RE tools have been evolving over time to meet the demand for flexibility, agile development, collaboration, and new ways of requirements management. De Gea et al. [20] carried out a survey to identify the existing RE tools and compared them featurewise. In spite of many existing specialized RE tools, Forward et al. [16] reported that most preferred tools for creating artifacts include word processors and text editors. Limited investigation has been done on the reasons why RE tools are not used frequently. Karlsson et al. [27] reported that one of the small companies that they have studied requested a simple RE tool since the existing ones would have a too large introduction overhead and a too steep learning curve. Hoffmann et al. [23] created a list of requirements of RE tools. The only proposed requirement in their work that addresses a challenge of our study is the ability to work collaboratively.

Almost 17 years ago, Myers et al. [40] predicted that a radical change in user interface design would occur. While we actually observed that UIs have matured since then, fundamental changes did not happen, especially in the tools that are used for requirements artifacts. Consequently, there is room for improvement in the UI of RE tools. Based on the related works that we reviewed about empirical studies on requirements artifacts and tools, there is no clear evidence in the literature to what extent working with RE artifacts is challenging and how successfully RE tools employed interaction techniques to enhance working with artifacts including creating, manipulating and managing artifacts. Our study contributes to filling this knowledge gap, thus providing an empirical basis for identifying the interaction requirements of RE tools.

8 Threats to validity

We identified the threats to the validity of our research and tried to minimize their effect on the final result. Below we discuss the usual four categories of validity [60].

Conclusion validity refers to finding a relation between data if it exists. Measurement reliability can affect conclusion validity. Therefore, to make our measures clear for the participants, we described our measures in detail and in a step-by-step manner. In addition, all interviews were conducted by the first author. When we needed a new measure, we defined it by combining other well-known measures (e.g., screen size and artifact size). We verified the consistency of our measures by asking duplicate questions. Moreover, we discussed the questions with RE experts and did a pilot study to avoid misunderstandings. Since we asked the participants to imagine having a screen size that they did not have in reality, still the accuracy of the gathered data depends on how accurately they can imagine that situation.

We made assumptions about what kind of people are more likely to have experienced challenges. Different assumptions might have resulted in different orders in our lists. Furthermore, we tried to have the interviews and surveys in similar conditions for the participants. We suggested the participants spend around 10 min for the survey. We scheduled the interview meetings in advance and asked the participants to be in a non-disruptive environment. Using statistical computations can be a threat to the validity of the results especially on lower sample sizes. In this study, we used such computations for prioritizing the challenges and not for identifying or filtering the challenges. In addition, we did not distinguish the practitioners who wrote artifacts and the practitioners who only read artifacts. This factor can affect the prioritization of the challenges.

Internal validity of an interview refers to making sure that the differences in the answers received are only because of the known differences among participants. Questions remained the same during the whole duration of the study. All 29 interviews were performed within a relatively short period of two months to avoid any software or hardware technology advancement. All participants were self-motivated and we did not offer any compensation.

Construct validity ensures that questions actually ask what they are supposed to ask. For example, we cannot guarantee that the participants remember everything related to our questions during the interview. So if a participant did not mention a challenge, this does not necessarily mean that they did not face that challenge. Therefore, in the analysis phase, we tried to minimize the influence of the frequency of the answers by prioritizing the challenges based on how much the participants were challenged. Moreover, although we did not have any hypothesis or expectation about the results, we were careful not to let the participants guess any hypothesis or expectation by mistake. For instance, we chose neutral tone in the interviews and ordered the questions of the survey randomly.

Our goal in this study was to investigate how practitioners work with RE artifacts, the challenges they face, and the related properties of artifacts. Eventually, we studied software engineering and RE artifacts together. The reason was that the challenges of working with requirements artifacts are not confined to RE, and recruiting participants that purely work with RE artifacts or can isolate their experience with RE artifacts from other artifacts was not possible. To assess this threat, later in the study, we compared the results from the participants who worked mostly with RE artifacts and the results from the participants who worked with more software engineering artifacts. The comparison did not show any statistically significant difference that we could report as a finding. Therefore, we conclude that our findings are valid for RE artifacts.

There are two threats concerning the ability of zooming. First, we did not investigate how much of an artifact is needed for a given task. It is possible that only a part of a large artifact is sufficient for carrying out a task. In this case, despite having an artifact that does not fit the screen, some of the challenges that we found may not apply. However, even in such a case, the practitioner has to scroll to find the needed part and adjust the zoom level to make it fit the screen. Second, we assumed that the resolutions of participants' screens are high enough to show the details of an artifact adequately when zoomed out extensively and the limitation for zooming out without making details unreadable is imposed by human eyesight only, not by the resolution of the screens. We made this assumption to reduce the duration and complexity of the interviews, considering that the probability of practitioners using outdated hardware is not high. In addition, to increase the accuracy of the measurement and make the participants more relaxed, we informed them that the data will be used and presented anonymously. We gathered information from various sources to avoid mono-operation bias.

External validity of a research means that the results are generalizable. For this purpose, the selected sample (the interviewees in our case) should not have certain features in common. This is very hard to achieve in an interview-based study. Some features were inevitably shared by all participants such as being volunteers who are interested in contributing to a scientific study and are social enough to answer our e-mail and participate in a one hour long interview. To avoid bias, we defined our criteria for selecting participants as simple as possible and used two different types of sampling. The variety of our final sample in terms of country, roles and company size shows that we were successful. Nevertheless, we cannot claim that our sample of 29 practitioners is statistically representative of the whole software development community. Consequently, as stated at the beginning of Sect. 4, the quantitative figures we report in our findings are not statistically generalizable. Hence, with respect to generalizability, our quantitative findings should be considered as hypotheses, rather than generally valid facts.

9 Conclusion and future work

In this paper, we presented a study about working with requirements and software development artifacts. We considered the properties of artifacts and tools that are related to information presentation and interaction. We also investigated the challenges related to these properties and the workarounds that practitioners employ to overcome them. Our goal was to gain an in-depth understanding of the state of practice in this area. To achieve this goal, we interviewed 29 practitioners from different companies located in eleven countries.

Our findings clarify the relations between the mentioned properties of artifacts, the challenges related to them, and how these challenges are handled in practice. We found that practitioners work with artifacts that are larger than their screens and with interconnected artifacts that have to be accessed simultaneously. In addition, they need to collaboratively work on their artifacts and keep the relationship information of the artifacts. Since the existing software tools do not provide sufficient support for conveniently carrying out such tasks, the practitioners we interviewed try to address the challenges encountered in various ways. For instance, they heavily rely on their memory or use other methods that are inefficient and

frequently error-prone, e.g., taking screenshots to remember, using paper and whiteboards to create artifacts, and using file name conventions to track the relations between artifacts.

Furthermore, our analysis of how our findings relate to other studies that address the challenges we identified encourages researchers and tool developers to study the reasons why these challenges exist in spite of the existing solutions.

Requirements engineers are strongly involved in creating and managing artifacts. So when framing our work in the context of RE, we can state that our results contribute to a better understanding of the challenges that requirements engineers face in working with requirements artifacts. Addressing these challenges is one way of improving RE tools. More efficient and more effective tools will enable requirements engineers to work with artifacts with less effort.

As a next step, we plan to use the results of this study to develop new approaches that enable practitioners to handle challenging artifact types, such as large artifacts or concurrently used sets of interconnected artifacts, in an efficient and effective way. We also envisage further studies focusing on individual challenges or a group of related challenges to discover specific requirements for future tools, e.g., by observing a team of requirements engineers when they collaboratively work on artifacts or inspecting the artifact interrelationship information they keep and store.

Acknowledgements The authors thank Dr. Irina Todoran Koitz for the insightful discussions and valuable feedback. We are grateful to the anonymous reviewers for helpful suggestions that enabled us to improve this paper. We would also like to thank all the participants and their companies for their time and valuable contributions that made the data collection possible for this research.

References

1. Begel A, Khoo YP, Zimmermann T (2010) Codebook: discovering and exploiting relationships in software repositories. In: Proceeding of the 32nd international conference on software engineering (ICSE 2010), vol 1, pp 125–134
2. Bennett C, Myers D, Storey MA, German DM, Ouellet D, Salois M, Charland P (2008) A survey and evaluation of tool features for understanding reverse-engineered sequence diagrams. J Softw Maint Evol Res Pract 20(4):291–315
3. Braun E, Amyot D, Lethbridge TC (2015) Generating software documentation in use case maps from filtered execution traces. In: 17th international SDL forum of model-driven engineering for smart cities. Springer, New York, pp 177–192
4. Cleland-Huang J, Habrat R (2007) Visual support in automated tracing. In: Proceeding of the second international workshop on requirements engineering visualization (REV 2007)
5. Cockburn A, Karlson A, Bederson BB (2009) A review of overview+detail, zooming, and focus+context interfaces. ACM Comput Surv 41(1):1–31
6. Cooper Jr JR, Lee SW, Gandhi RA, Gotel O (2009) Requirements engineering visualization: a survey on the state-of-the-art. In: Proceeding of the fourth international workshop on requirements engineering visualization (REV 2009), pp 46–55
7. Cornelissen B, Van Deursen A, Moonen L, Zaidman A (2007) Visualizing testsuites to aid in software understanding. In: Proceeding of the 11th European conference on software maintenance and reengineering (CSMR'07), pp 213–222
8. Creswell JW (2013) Research design: qualitative, quantitative, and mixed methods approaches. Sage Publications, Thousand Oaks
9. Czerwinski M, Smith G, Regan T, Meyers B, Robertson G, Starkweather, G (2003) Toward characterizing the productivity benefits of very large displays. In: Proceedings of the IFIP TC13 international conference on human-computer interaction (INTERACT '03), vol 3. IOS Press, Amsterdam, pp 9–16
10. Damian D, Chisan J (2006) An empirical study of the complex relationships between requirements engineering processes and other processes that lead to payoffs in productivity, quality, and risk management. IEEE Trans Softw Eng 32(7):433–453
11. Davis A, Dieste O, Hickey A, Juristo N, Moreno AM (2006) Effectiveness of requirements elicitation techniques: empirical results derived from a systematic review. In: 14th IEEE international requirements engineering conference (RE '06), pp 179–188
12. De Gea JMC, Nicolás J, Alemán JLF, Toval A, Ebert C, Vizcaíno A (2012) Requirements engineering tools: capabilities, survey and assessment. Inf Softw Technol 54(10):1142–1157
13. Dekel U (2005) Supporting distributed software design meetings: what can we learn from co-located meetings? In: Proceedings of the 2005 workshop on human and social factors of software engineering (HSSE '05), pp 1–7
14. Dekel U, Herbsleb JD (2007) Notation and representation in collaborative object-oriented design: an observational study. In: ACM SIGPLAN Notices, vol 42. ACM, New York, pp 261–280
15. Easterbrook S, Singer J, Storey MA, Damian D (2008) Selecting empirical methods for software engineering research. In: Guide to advanced empirical software engineering. Springer, New York, pp 285–311
16. Forward A, Lethbridge TC (2002) The relevance of software documentation, tools and technologies: a survey. In: Proceedings of the 2002 ACM symposium on document engineering (DocEng '02). ACM, New York, pp 26–33
17. Frisch M, Dachselt R (2010) Off-screen visualization techniques for class diagrams. In: Proceedings of the 5th international symposium on software visualization (SOFTVIS '10). ACM, New York, pp 163–172
18. Frøkjær E, Hertzum M, Hornbæk K (2000) Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '00). ACM, New York, pp 345–352
19. Furnas GW (1986) Generalized fisheye views. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '86). ACM, New York, pp 16–23
20. de Gea JMC, Nicolás J, Alemán JLF, Toval A, Ebert C, Vizcaíno A (2011) Requirements engineering tools. IEEE Softw 28(4):86–91
21. Ghazi P, Glinz M (2016) An exploratory study on user interaction challenges when handling interconnected requirements artifacts of various sizes. In: Proceeding of the 24th IEEE international requirements engineering conference (RE '16), pp 76–85
22. Ghazi P, Seyff N, Glinz M (2015) FlexiView: A magnet-based approach for visualizing requirements artifacts. In: Proceeding of the 21st international working conference on requirements engineering: foundation for software quality (REFSQ '15). Springer, New York, pp 262–269

23. Hoffmann M, Kuhn N, Weber M, Bittner M (2004) Requirements for requirements management tools. In: Proceedings of 12th IEEE international requirements engineering conference (RE '04), pp 301–308
24. IEEE: IEEE standard glossary of software engineering terminology. IEEE Std 610.12.1990, pp 1–84 (1990)
25. International Organization for Standardization: ISO 9241-11: Ergonomic requirements for office work with visual display terminals (VDTs): Part 11—guidance on usability (1998)
26. Kagdi H, Maletic JI (2007) Onion graphs for focus+context views of UML class diagrams. In: 4th IEEE international workshop on visualizing software for understanding and analysis (VISSOFT 2007), pp 80–87
27. Karlsson L, Dahlstedt Å, Natt och Dag J, Regnell B, Persson A, (2002) Challenges in market-driven requirements engineering-an industrial interview study. In: Eighth international workshop on requirements engineering: foundation for software quality (REFSQ '02), pp 37–49
28. King N, Horrocks C (2010) Interviews in qualitative research. Sage, London
29. Kitchenham B, Pfleeger SL (2002) Principles of survey research part 5: populations and samples. ACM SIGSOFT Softw Eng Notes 27(5):17–20
30. Koning H, Dormann C, van Vliet H (2002) Practical guidelines for the readability of IT-architecture diagrams. In: Proceedings of the 20th annual international conference on computer documentation (SIGDOC '02). ACM, New York, pp 90–99
31. Lam H, Munzner T (2010) A guide to visual multi-level interface design from synthesis of empirical study evidence. Synth Lect Vis 1(1):1–117
32. Lethbridge TC, Sim SE, Singer J (2005) Studying software engineers: data collection techniques for software field studies. Empir Softw Eng 10(3):311–341
33. Lethbridge TC, Singer J, Forward A (2003) How software engineers use documentation: the state of the practice. IEEE Softw 20(6):35–39
34. Likert R (1932) A technique for the measurement of attitudes. Arch Psychol 140:1–55
35. Lischke L, Mayer, S, Wolf K, Henze N, Schmidt A, Leifert S, Reiterer H (2015) Using space: effect of display size on users' search performance. In: Proceedings of the 33rd annual ACM conference extended abstracts on human factors in computing systems (CHI '15). ACM, New York, pp 1845–1850
36. Liskin O (2015) How artifacts support and impede requirements communication. In: International working conference on requirements engineering: foundation for software quality (REFSQ '15). Springer, New York, pp 132–147
37. Mangano N, Baker A, Dempsey M, Navarro E, van der Hoek A (2010) Software design sketching with Calico. In: Proceedings of the international conference on automated software engineering (ASE '10). ACM, New York, pp 23–32
38. Martinez WL, Martinez AR (2007) Computational statistics handbook with MATLAB. CRC Press, Boca Raton
39. Müller SC, Fritz T (2013) Stakeholders' information needs for artifacts and their dependencies in a real world context. In: Proceedings of the 29th IEEE international conference on software maintenance (ICSM '13), pp 290–299
40. Myers B, Hudson SE, Pausch R (2000) Past, present, and future of user interface software tools. ACM Trans Comput Human Interact 7(1):3–28
41. Oates BJ (2005) Researching information systems and computing. Sage, London
42. Ossher H, van der Hoek A, Storey MA, Grundy J, Bellamy R (2010) Flexible modeling tools (Flexitools 2010). In: Proceedings of the 32nd international conference on software engineering (ICSE '10). ACM, New York, vol 2, pp 441–442
43. Ossher H, van der Hoek A, Storey MA, Grundy J, Bellamy R, Petre M (2011) Workshop on flexible modeling tools (Flexitools 2011). In: Proceedings of the 33rd international conference on software engineering (ICSE '11). ACM, New York, pp 1192–1193
44. Pagano D, Brügge B (2013) User involvement in software evolution practice: a case study. In: Proceedings of the 35th international conference on software engineering (ICSE '13), pp 953–962
45. Parsons P (2013) Cognitive activity support tools: design of the visual interface. Ph.D. thesis, The University of Western Ontario
46. Pohl K (2010) Requirements engineering: fundamentals, principles, and techniques. Springer Publishing Company, Incorporated
47. Portillo-Rodríguez J, Vizcaíno A, Piattini M, Beecham S (2012) Tools used in global software engineering: a systematic mapping review. Inf Softw Technol 54(7):663–685
48. Reddivari S, Chen Z, Niu N (2012) ReCVisu: a tool for clustering-based visual exploration of requirements. In: Proceeding of the 20th IEEE international requirements engineering conference (RE '12), pp 327–328
49. Reinhard T, Meier S, Glinz M (2007) An improved fisheye zoom algorithm for visualizing and editing hierarchical models. In: Proceeding of the second international workshop on requirements engineering visualization (REV 2007)
50. Reinhard T, Meier S, Stoiber R, Cramer C, Glinz M (2008) Tool support for the navigation in graphical models. In: Proceeding of the 30th international conference on software engineering (ICSE '08). ACM, New York, pp 823–826
51. Robertson G, Czerwinski M, Baudisch P, Meyers B, Robbins D, Smith G, Tan D (2005) The large-display user experience. IEEE Comput Graph Appl 25(4):44–51
52. Robson C (2002) Real world research, 2nd edn. Blackwell Publishing, Malden
53. Seaman CB (2008) Qualitative methods. In: Guide to advanced empirical software engineering. Springer, New York, pp 35–62
54. Sommerville I (1992) Documentation. Software Engineering, 4th edn. Addison-Wesley, USA, pp 571–587
55. de Souza CR, Redmiles DF (2008) An empirical study of software developers' management of dependencies and changes. In: Proceedings of the 30th international conference on software engineering (ICSE '08). ACM, New York, pp 241–250
56. de Souza SCB, Anquetil N, de Oliveira KM (2005) A study of the documentation essential to software maintenance. In: Proceedings of the 23rd annual international conference on design of communication: documenting and designing for pervasive information. ACM, New York, pp 68–75
57. Teruel MA, Navarro E, López-Jaquero V, Montero F, González P (2014) A CSCW requirements engineering case tool: development and usability evaluation. Inf Softw Technol 56(8):922–949
58. Winkler S (2007) Information flow between requirement artifacts. results of an empirical study. In: International working conference on requirements engineering: foundation for software quality (REFSQ '07). Springer, New York, pp 232–246
59. Winkler S, Pilgrim J (2010) A survey of traceability in requirements engineering and model-driven development. Softw Syst Model 9(4):529–565
60. Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B, Wesslén A (2000) Experimentation in software engineering: an introduction. Kluwer Academic Publishers, Boston
61. Wüest D, Seyff N, Glinz M (2012) Flexible, lightweight requirements modeling with FlexiSketch. In: Proceedings of the 20th IEEE international requirements engineering conference (RE '12), pp 323–324
62. Wüest D, Seyff N, Glinz M (2012) FlexiSketch: a mobile sketching tool for software modeling. In: Proceedings of the fourth international conference on mobile computing,

applications and services (MobiCASE). Springer, New York, pp 225–244
63. Wüest D, Seyff N, Glinz M (2015) Sketching and notation creation with FlexiSketch Team: evaluating a new means for collaborative requirements elicitation. In: Proceeding of the 23rd IEEE international requirements engineering conference (RE '15), pp 186–195
64. Zhi J, Garousi-Yusifoğlu V, Sun B, Garousi G, Shahnewaz S, Ruhe G (2015) Cost, benefits and quality of software development documentation: a systematic mapping. J Syst Softw 99:175–198